Grado en Ingeniería Informática

Trabajo Final de Grado

Desarrollo de Biblioteca y Red en una plataforma de educación basada en Symfony

Autor:
Javier Martínez de la Torre

Supervisor:
Héctor Saiz Sánchez
Tutor académico:
Begoña Martínez Salvador

Fecha de lectura: 26 de Junio de 2017
Curso académico 2016/2017

Resumen

Esta memoria y desarrollo del proyecto forman parte del proyecto de fin de grado del Grado de Ingeniería Informática en la Universidad Jaume I. La estancia en prácticas se llevó a cabo en la empresa CIDET, una empresa que se dedica a la educación y se encuentra en la parque tecnológico, Espaitec, dentro de Universidad Jaume I.

En este documento se describe el proyecto de desarrollo de dos funcionalidades nuevas, una donde los usuarios puedan guardar y añadir ficheros, llamada Biblioteca. Y otra donde puedan crear grupos de usuarios para intercambiar mensajes y puedan compartir mensajes con otros usuarios que sean sus amigos, esta funcionalidad se llama Red. Estas dos nuevas funcionalidades se añaden a la plataforma llamada Edueca (www.edueca.com). Este proyecto se desarrolla utilizando Symfony 2 ya que todo Edueca se desarrolla sobre Symfony 2. En esta memoria primero se mostrará el análisis, diseño y la documentación sobre la plataforma y el software a utilizar, más tarde se mostrará el proceso realizado para llevar a cabo el proceso mediante Symfony 2, JavaScript, PHP, HTML ,CSS y otros tipos de lenguajes y herramientas.

Palabras clave

Edueca, Red, Biblioteca, Symfony, PHP

Keywords

Edueca, Net, Library, Symfony, PHP

Índice general

Índice de figuras

Capítulo 1

Introducción

Esta memoria presenta el trabajo realizado durante la estancia en prácticas en la empresa *Centre for the Innovation and Development of Education and Technology*. El proyecto realizado añade funcionalidades a la plataforma Edueca, que se describe en el capítulo 1.2. En este primer capítulo se va a desarrollar la descripción del contexto de la empresa y la motivación del proyecto. En el siguiente capítulo se narrará la situación previa de la plataforma Edueca y las nuevas herramientas con las que cuenta tras la estancia en prácticas. En el tercer capítulo se presenta la planificación del proyecto, se describe la metodología Scrum, las historias de usuario y la estimación de recursos para este proyecto. Durante el cuarto capítulo se desarrolla el análisis y diseño del proyecto donde se muestran los distintos tipos de diagramas y los prototipos. Por último se desarrolla el capítulo de implementación donde se describe el trabajo realizado en cada historia de usuario. En la parte final del documento existen dos anexos, el primero realiza una breve explicación general de cómo trabajar con Symfony y el segundo describe algunas de las herramientas utilizadas en el proyecto.

1.1. Contexto y motivación del proyecto

El proyecto se ha desarrollado en la empresa CIDET (ver figura 1.1) durante la estancia en prácticas, en la cual diseñé e implementé nuevas herramientas en la plataforma Edueca, plataforma que están desarrollando en dicha empresa (ver capítulo 1.2). CIDET se trata de una empresa que se encuentra en el parque tecnológico, Espaitec, dentro de Universidad Jaume I.

Figura 1.1: *Logotipo de Cidet*

CIDET es una empresa que se dedica a la educación. Los servicios que ofrece esta empresa se dividen en tres áreas:

- *cidet Learning lab* (figura 1.2): ofrecen actividades formativas dirigidas a adultos y personas mayores.
  - En temas tecnológicos, pero también sociales, culturales y humanos.
  - Dentro y fuera del aula con metodologías formales e informales.
  - En espacios físicos y también virtuales (on-line).

Figura 1.2: *Logo de Learning lab*

- *Educational Technovation* (figura 1.3): ponen en común la tecnología y la educación para que las personas y las organizaciones puedan crecer.
  - Proyectos de cooperación KA1 Erasmus+ (`http://sepie.es/educacion-escolar/seguimiento-2016-KA1.html`) de la Unión Europea.
  - Creación de entornos de aprendizaje a medida.
  - Diseño y creación de materiales formativos.
  - Formación de formadores, conferencias y eventos.

Figura 1.3: *Logo de Technovation*

- *Edueca* (figura 1.4): desarrollo de una plataforma para proyectos educativos y de crecimiento. Sus características son:
  - Diseñado para adultos; fácil de manejar, colaborativo y divertido.
  - Rápido y ágil de gestionar y organizar por parte del profesor.
  - Capaz de crecer e innovar con tu equipo o grupo de trabajo.
  - Añadiendo valor y conocimiento a tu organización o empresa.

Es en la última parte, *Edueca*, donde se centra mi proyecto de estancia en prácticas. Se pretende que los usuarios de esta plataforma cuenten con un apartado donde poder guardar contenidos de cursos, añadir ficheros propios, etc. Este es el apartado conocido como Biblioteca. Además de la Biblioteca, se pretende realizar un apartado llamado Red que incluya una funcionalidad mucho más social a la plataforma de forma que un usuario pueda publicar y compartir mensajes y contenido con aquellos usuarios que sean amigos suyos. También incluirá opciones para crear grupos de usuarios, crear un grupo de tutorías con un profesor, etc.

Figura 1.4: *Logo de Edueca*

1.2. Edueca

Edueca es una plataforma on-line que facilita el aprendizaje y crecimiento a los usuarios, ya sean alumnos, formadores, equipos de trabajo o una organización. Contiene las siguientes funcionalidades:

- Permite generar cursos a los que añadir contenidos y estructurarlos por temas, como muestra la figura 1.5. Estos contenidos pueden contener texto y ficheros como vídeos, pdfs, imágenes, etc. Una vez generados los usuarios que pertenezcan al proyecto pueden visualizarlos como muestra la figura 1.6.

- Un curso puede contener un foro en el cual los usuarios del curso o proyecto pueden publicar mensajes y con ellos generar conversaciones con el resto de usuarios del curso (ver figura 1.7).

- Un curso o proyecto puede contener un Blog donde publicar noticias, ideas, pensamientos o descubrimientos (ver figura 1.8).

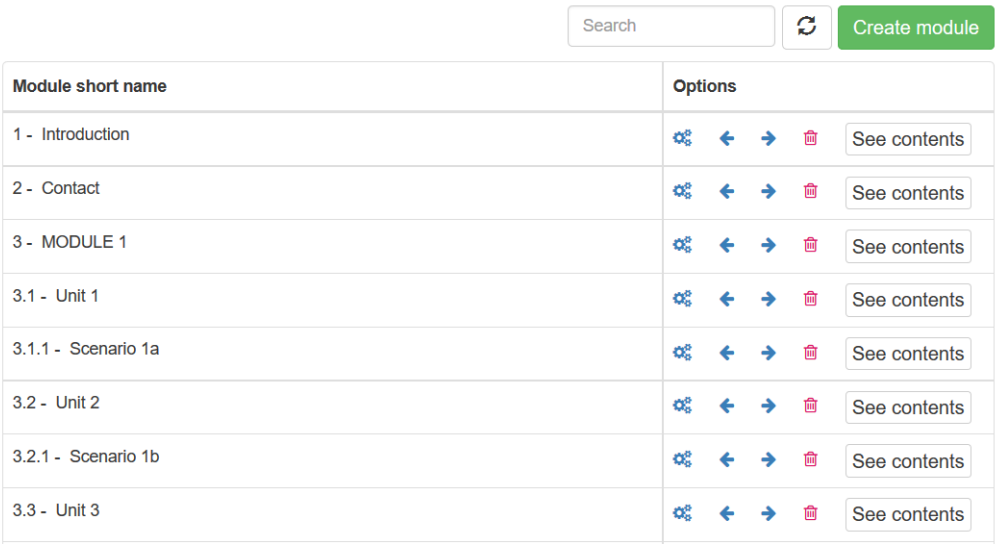

Figura 1.5: *Imagen de la generación de contenidos*

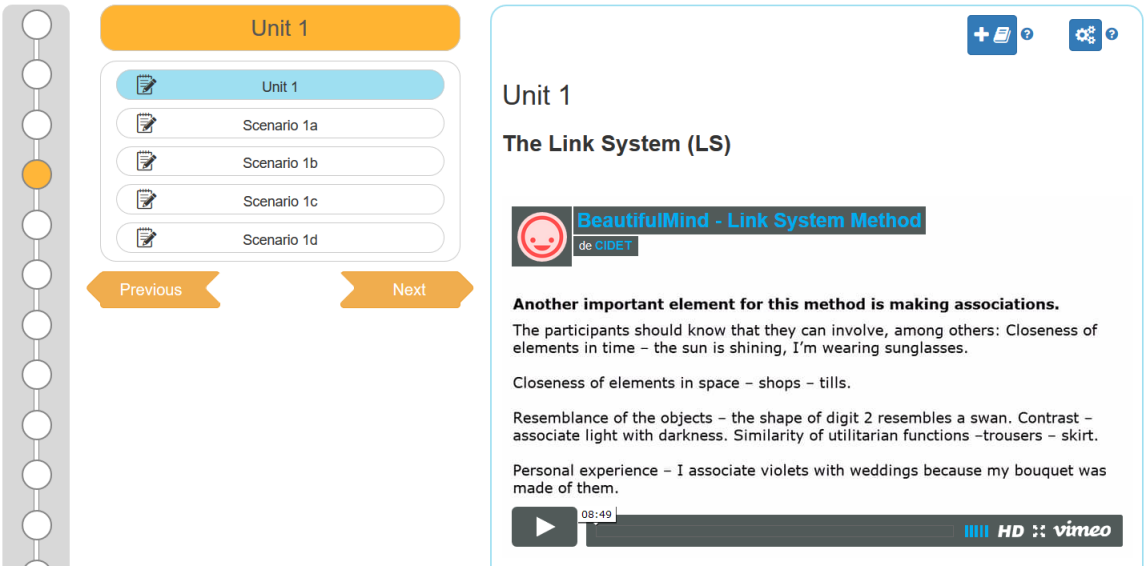

Figura 1.6: *Visualización de un contenido de un proyecto o curso*

New topic

Title	Author	Replies	Last message
CENA NAVIDAD	loli	11	on 09-12-16
cena de navidad	al366246	2	on 13-11-16
Nube	Lazi	1	on 10-11-16
libro de filosofia	Vicente	7	on 09-11-16
filosofia	al366246	1	on 09-11-16
cena de navidad	al366246	0	on 07-11-16
Acertijo	Lazi	4	on 04-11-16
INFORMACIÓN	Salud	1	on 03-11-16
Recogida de setas	Lazi	2	on 30-10-16
Bienvenidos al foro !!! ¿Cómo escribir?	Roger	5	on 30-10-16

Figura 1.7: *Foro de un curso de Edueca*

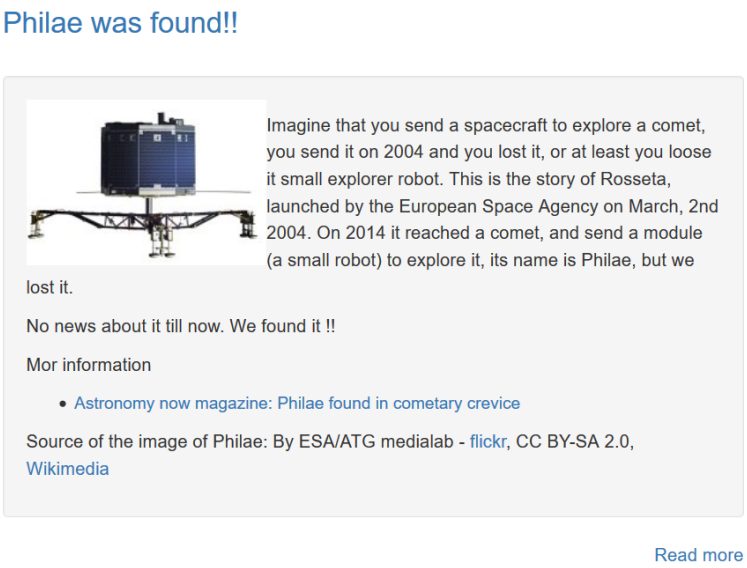

Figura 1.8: *Blog de un curso de Edueca*

1.3. Objetivos del proyecto

Los objetivos de este proyecto incluyen el desarrollo de dos nuevas funcionalidades en la plataforma, la Biblioteca y la Red. Se pretende que estos apartados sean fáciles de utilizar por los usuarios por lo que deberá probarse el funcionamiento con usuarios de prueba.

Las características que debe cumplir la funcionalidad de Biblioteca son:

- Permitir a los usuarios subir ficheros a la Biblioteca y descargar ficheros.
- Permitir añadir y quitar etiquetas a los ficheros.
- Mostrar información sobre los ficheros del usuario.
- Permitir generar un PDF y añadirlo a la Biblioteca a partir de un contenido de un proyecto como puede ser un tema de un curso.

Las características que debe cumplir la funcionalidad de Red son:

- Permitir a los usuarios crear grupos y añadir usuarios a dicho grupo.
- Permitir a los usuarios crear un grupo de tutoría con un profesor.
- Generar un perfil para cada usuario donde publicar mensajes y que sean visibles por aquellos usuarios que lo tengan como amigo.
- Permitir a los usuarios buscar otros usuarios y visitar sus muros.

Capítulo 2

Descripción del proyecto

2.1. Situación previa

La plataforma Edueca cuenta con una serie de características previas a la realización de la Biblioteca y la Red. Estas características o herramientas son: la creación de cursos o proyectos, módulos, foros y álbumes, permitir a los usuarios registrarse, iniciar sesión, recuperar su contraseña en caso de olvido y registrarse en un curso, por último permite invitar alumnos a un curso, añadir una caducidad a un contenido de un curso y añadir una imagen como logo de una institución.

2.2. Sistema tras el proyecto

En este apartado se van a describir con más detalle las funcionalidades que se han añadido a Edueca para ampliar las características anteriormente descritas de la plataforma. Esta plataforma se encuentra disponible en el servidor de la empresa y se puede acceder mediante el siguiente enlace `https://www.edueca.com`.

El proyecto se divide en dos apartados: la Biblioteca y la Red.

2.2.1. Biblioteca

El apartado de Biblioteca se divide en dos pantallas:

- Una pantalla donde poder subir ficheros y que de la opción de añadirles etiquetas antes de subirlos.
- Otra pantalla donde poder ver y mantener los ficheros que el usuario ha añadido a la biblioteca.

Este apartado contará con las siguientes funcionalidades:

- Filtrar ficheros según las etiquetas que tengan.
- Añadir ficheros a la plataforma.
- Eliminar etiquetas de un fichero o eliminar una misma etiqueta de todos los ficheros que la contengan.
- Cambiar el nombre de los ficheros.
- Borrar y descargar un fichero.
- Añadir las mismas etiquetas a varios ficheros.
- Visualizar PDF's, documentos de texto e imágenes.
- Buscar ficheros por nombre.

2.2.2. Red

Por otro lado el apartado de Red se divide en las siguientes pantallas:

- Una pantalla donde se muestran todos los mensajes de los amigos del usuario y permite publicar un mensaje y buscar usuarios.
- Una pantalla llamada muro donde se muestran solo los mensajes publicados por el usuario y las respuestas a los mismos.
- Pantalla con el contenido de los diferentes grupos del usuario.

Este apartado contará con las siguientes funcionalidades:

- Publicar mensajes en la pantalla del muro del usuario y en la pantalla de mensajes de amigos.
- Realizar comentarios sobre los todos mensajes de usuario que aparezcan en la plataforma.
- Marcar un mensaje como "me gusta".
- Desmarcar un "me gusta" sobre un mensaje sobre el que previamente el usuario había marcado "me gusta".
- Buscar el muro de otros usuarios para añadirlos como amigos.
- Visualizar los grupos a los que pertenezca el usuario.
- Generar grupos y añadir a otros usuarios a los grupos.

- Abandonar un grupo al que pertenezca el usuario.
- Publicar mensajes y comentarios en un grupo.
- Visualizar usuarios pertenecientes a un grupo donde se encuentre el usuario.

Capítulo 3

Planificación del proyecto

3.1. Metodología

El proyecto de la estancia en prácticas se ha desarrollado utilizando la metodología Scrum [16]. En Scrum un proyecto se ejecuta en iteraciones de duración fija llamadas sprints. Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

La duración de un sprint es normalmente de 2 semanas, aunque en algunos equipos son de 3 o 4 semanas, límite máximo de feedback y reflexión. En cada sprint se realiza una reunión con el equipo para aclarar qué se va desarrollar en dicho sprint.

El proceso comienza con la elaboración del Product Backlog. Se trata de un fichero que contiene el conjunto de tareas, los requerimientos y las funcionalidades requeridas por el proyecto. Cualquier miembro del equipo puede modificar este documento pero el único con autoridad para agregar prioridades es el Product Owner.

Existen diferentes perfiles en la metodología Scrum:

- Product Owner: representa la voz del cliente y del resto de interesados no implicados directamente en el proyecto. Se encarga de definir los objetivos del proyecto y participa en las reuniones de los sprints para revisar los requisitos completados.
- Scrum Master: asegura que el resto del equipo no tiene problemas para abordar sus funciones y tareas. Este perfil ayuda al equipo a mantenerse activo y productivo.
- Scrum Team: es el equipo encargado de desarrollar y entregar el producto.
- Stakeholders: comprende el resto de perfiles interesados en el producto: directores, dueños, comerciales, etc.

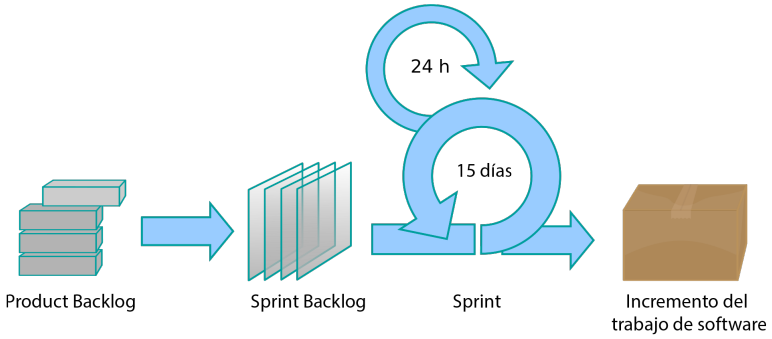

Figura 3.1: *Funcionamiento de Scrum*

3.2. Planificación

Para desarrollar los apartados de la Biblioteca y Red se han definido las diferentes historias de usuario a las que se les ha asignado story points [19]. Los story points son una medida arbitraria utilizada por los equipos Scrum. Se usan para medir el esfuerzo requerido para implementar una historia de usuario.

- **HU-01 Pantalla para visualizar ficheros**

  Realizar un apartado para que el usuario pueda ver sus ficheros. *Story point: 5*

- **HU-02 Subir ficheros**

  Como usuario quiero poder subir un fichero especificando un nombre nuevo y añadir etiquetas sobre el mismo antes de subirlo. *Story point: 8*

- **HU-03 Borrar ficheros**

  Como usuario quiero poder borrar mis archivos. *Story point: 2*

- **HU-04 Subir ficheros arrastrándolos**

  Como programador quiero añadir un *dropzone* en la zona de los archivos para poder arrastrar un archivo y que se suba a la plataforma. *Story point: 2*

- **HU-05 Gestionar etiquetas en los ficheros**

  Como usuario quiero poder añadir y quitar etiquetas de un archivo para poder buscar los archivos por etiquetas. *Story point: 3*

- **HU-06 Descargar de la biblioteca**

  Como usuario quiero poder descargar los archivos de la biblioteca. *Story point: 2*

- **HU-07 Añadir etiqueta a muchos ficheros**

  Como usuario quiero poder añadir la misma etiqueta a los ficheros seleccionados para agilizar el añadir la misma etiqueta a muchos archivos. *Story point: 2*

- **HU-08 Buscar ficheros por etiquetas**

  Como usuario quiero poder buscar ficheros por etiquetas para agilizar la navegación entre mis ficheros. *Story point: 3*

- **HU-09 Cambiar nombre de fichero**

  Como usuario quiero poder cambiar el nombre de un fichero. *Story point: 2*

- **HU-10 Borrar una etiqueta de todos los ficheros**

  Como usuario quiero poder borrar una etiqueta en el menú de etiquetas para que se borre de todos mis ficheros. *Story point: 3*

- **HU-11 Ver imagen**

  Como usuario quiero poder visualizar la imagen si al seleccionar un fichero es una imagen. *Story point: 5*

- **HU-12 Añadir fichero de la biblioteca a un contenido de un curso o proyecto**

  Como profesor o creador de un proyecto se podrá añadir un fichero de la biblioteca mientras se está creando un contenido de un curso o proyecto. *Story point: 5*

- **HU-13 Generar PDF a partir de contenido**

  Como usuario quiero poder añadir a la biblioteca un pdf generado a partir de un fichero o un texto que se encuentra en un proyecto. *Story point: 8*

- **HU-14 Redirigir a la pantalla de login**

  Como usuario quiero que si no estoy identificado al entrar en la biblioteca entonces, se redirija a la pantalla de identificación o registro. *Story point: 2*

- **HU-15 Mostrar tamaño consumido**

  Como programador quiero que se muestre en la parte baja de la biblioteca el tamaño total consumido en el sistema por los ficheros de ese usuario. *Story point: 3*

- **HU-16 Pantalla de Muro**

  Como usuario quiero un muro personal donde poder publicar mensajes y que otros usuarios puedan comentar a esos mensajes. *Story points:* 8.

- **HU-17 Mensajes de amigos**

  Como usuario quiero un apartado donde visualizar los mensajes de mis usuarios "amigos". *Story points:* 5.

- **HU-18 Buscar usuarios**

  Como usuario quiero poder buscar a otros usuarios. *Story points:* 8.

- **HU-19 Añadir a amigos**

  Como usuario quiero poder añadir a otro usuario como amigo con el fin de que sus publicaciones sean visualizadas en mi apartado de amigos. *Story points:* 3.

- **HU-20 Eliminar amigo**

  Como usuario quiero poder eliminar un amigo. *Story points:* 3.

- **HU-21 Visualizar PDF o fichero de texto**

  Como usuario quiero poder visualizar el contenido de un pdf o de un fichero de texto desde la biblioteca. *Story points: 8.*

3.3. Estimación de recursos

3.3.1. Recursos software

En este proyecto se utilizará PHP[14], Javascript [10], HTML [9], CSS [5], PhpStorm [15], Twig [23], JIRA [11], MySQL [13], Bitbucket [2], Symfony [20] y Bootstrap [3] para el desarrollo del código de los distintos apartados.

PHP

PHP es un acrónimo recursivo que significa PHP Hypertext Preprocessor (inicialmente PHP Tools, o, Personal Home Page Tools). Se trata de un lenguaje adecuado para el desarrollo web y que, junto con HTML, permite crear sitios web dinámicos. PHP se instala en el servidor y funciona con versiones de Apache, Microsoft IIs, Netscape Enterprise Server y otros.

El código PHP es insertado dentro del código HTML de un sitio web. Cuando un cliente visita la página web que contiene este código, el servidor lo ejecuta y el cliente sólo recibe el resultado. Su ejecución, es por tanto en el servidor, a diferencia de otros lenguajes de programación que se ejecutan en el navegador.

Fue creado originalmente por Rasmus Lerdorf en el año 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP. Este lenguaje forma parte del software libre publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU debido a las restricciones del uso del término PHP.

Javascript

Se trata de uno de los lenguajes de programación más conocidos, es un lenguaje interpretado y orientado a objetos que suele utilizarse para el desarrollo de scripts en páginas web.

Abreviado comúnmente como JS se utiliza principalmente en el lado del cliente, al contrario que PHP que se ejecuta en el lado del servidor, permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Desde 2012 todos los navegadores modernos soportan ECMAScript 5.1 [7], una versión de script. Actualmente se suele utilizar para el envio y recepción de datos del servidor mediante llamadas AJAX [1] (Asynchronous JavaScript And XML).

HTML

HTML (HyperText Markup Language) se trata de un lenguaje de marcado para la elaboración de páginas web. Es un lenguaje de marcado ya que incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación. Se basa en

el metalenguaje SMGL [18] (Standard Generalized Markup Languaje).

El lenguaje HTML nace en 1991 y lo crea Tim Berners-Lee como un sistema hipertexto con el único objetivo de servir como medio de transmisión de información entre los científicos que se ocupaban de la Física de alta energía, como parte de la iniciativa World Wide Web. HTML tiene una gran aceptación a partir de 1994 y desde entonces ha ido evolucionando hasta su version 5.1 que es la actual.

CSS

CSS (Cascading Style Sheets) se trata de un lenguaje para describir la presentación de documentos estructurados escritos en lenguaje de marcado. El código de CSS muestra como debe ser renderizado un elemento en la pantalla.

Junto con HTML y JavaScript, CSS es usado por muchos sitios web para crear páginas visualmente atractivas, interfaces de usuario para aplicaciones web, y GUIs para muchas aplicaciones móviles. Su principal función es mantener un mismo estilo en varios documentos HTML.

La especificación CSS es mantenida por el World Wide Web Consortium [24] (W3C). El MIME type [12] text/css está registrado para su uso por CSS descrito en el RFC 2318 (March 1998).

Symfony

Symfony es un framework diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón Vista Controlador. Symfony tiene como objetivo acelerar la creación y mantenimiento de aplicaciones web y sustituir las tareas de codificación repetitivas. Está desarrollado completamente con PHP y es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.

La primera versión de Symfony fue publicada en 2005 por Fabien Potencier y tras el éxito que tuvo en el desarrollo de una página web para comercio electrónico y algunos otros proyectos, decidió liberarlo bajo una licencia open source.

MySQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation, esta basado en lenguaje de consulta estructurado (SQL).

MySQL se ejecuta en prácticamente todas las plataformas, incluyendo Linux, UNIX y Windows. A pesar de que se puede utilizar en una amplia gama de aplicaciones, MySQL se asocia más con las aplicaciones basadas en la web y la publicación en línea.

PhpStorm

Durante la estancia en prácticas se utilizó PhpStorm, que es un entorno de desarrollo integrado, para poder realizar y depurar el código del proyecto en Symfony, Twig y PHP.

Twig

Se trata de un lenguaje de plantillas para PHP. Symfony viene con un Bundle (ver qué es un bundle en la introducción a Symfony en el Anexo 1) para soportar Twig.

JIRA

Jira es una herramienta web para la gestión y seguimiento de proyectos. Se utiliza para el desarrollo de proyectos de forma colaborativa, la herramienta utilizada durante la estancia en prácticas es la pizarra. La pizarra se trata de una herramienta donde los usuarios pueden añadir tareas, asignarlas a usuarios y hacer comentarios sobre ellas de forma que la tarea según si está sin hacer o terminada, se está testeando o está parada se encontrará en una columna de la pantalla. Estas columnas tienen los siguientes nombres: "TO DO", "TESTING", "STAND BY" y "DONE".

Bitbucket

Se trata de un servicio de alojamiento para los proyectos que utilizan el sistema de control de versiones Mercurial y Git, en este caso concreto en Git.

Bootstrap

Bootstrap es un framework de Javascript de código abierto desarrollado por el equipo de Twitter. Es una combinación de código HTML, CSS y Javascript diseñado para ayudar a crear componentes de interfaz de usuario. Bootstrap también fue programado para soportar HTML5 y CSS3.

Bootstrap es una colección gratuita de herramientas para crear sitios web y aplicaciones web. Además contiene plantillas de diseño basadas en HTML y CSS para tipografía, formularios, botones, navegación y otros componentes de interfaz, así como extensiones de JavaScript opcionales.

3.3.2. Recursos humanos

Las tareas descritas anteriormente van a ser desarrolladas por el alumno bajo la supervisión del supervisor de la empresa. Es por ello que el alumno tomará tanto el rol de analista como el de programador.

3.3.3. Recursos hardware

Respecto a los recursos hardware utilizados para el desarrollo del proyecto se ha utilizado una pantalla de 27" junto con un MacBook Air el cual cuenta con un procesador i5, 4 GB de RAM y 512 GB de almacenamiento y una pantalla de 13". Además del ordenador, en CIDET cuentan con un servidor en Amazon utilizando los servicios S3 (Simple Storage Service). En este servidor se encuentra la base de datos y Edueca.

3.4. Seguimiento del proyecto

El proyecto se ha dividido en Sprints según establece la metodología Scrum. Se han realizado 5 sprints, debido a que a cada uno de los sprints se le han asignado 18 Story points.

El control del desempeño de las tareas y que se avance según lo planeado lo ha realizado el supervisor de la empresa. Este ha realizado un seguimiento de las tareas efectuadas mediante el JIRA y era él quien añadía nuevas tareas en las reuniones contempladas por la metodología Scrum y una vez finalizadas testeaba su correcto funcionamiento. Durante estas reuniones también se producía una elección de las tareas que se realizarían en el siguiente sprint eligiendo de las tareas restantes del backlog.

El desarrollo de las historias de usuario se muestra en el *burn down chart* de la figura 3.2. En esta figura se observa la realización real de story points frente a al transcurso ideal de los story points del proyecto respecto al tiempo.

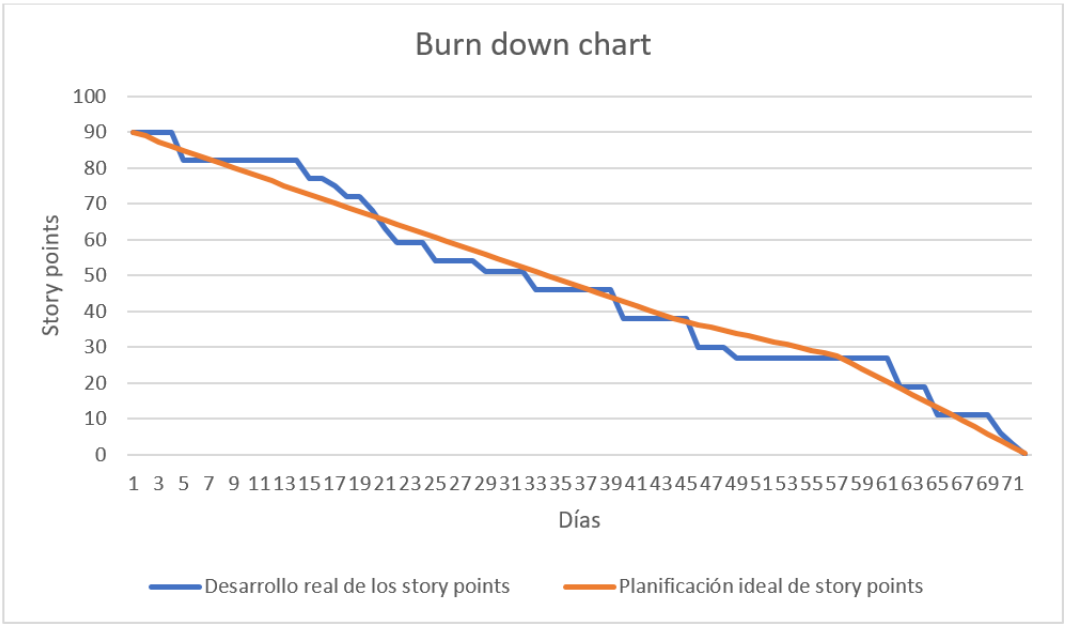

Figura 3.2: *Burn down chart del proyecto realizado*

Capítulo 4

Análisis y diseño del sistema

Este apartado detalla el análisis y diseño del proyecto empezando por las funcionalidades y requisitos que debe cumplir el sistema y los requisitos de datos funcionales. Más tarde se aborda el diseño del proyecto realizado y por último se muestra el diseño de la interfaz de las pantallas nuevas que este proyecto ha añadido a la plataforma Edueca.

4.1. Análisis del sistema

4.1.1. Requisitos funcionales

A continuación se detalla los distintos requisitos funcionales del proyecto. Estos requisitos funcionales muestran los servicios que el sistema ha de satisfacer, la manera en que éste debe reaccionar a entradas particulares y cómo se debe comportar en situaciones particulares. En ocasiones, también pueden declarar lo que el sistema no debe hacer.

Los requisitos de comportamiento para cada requisito funcional se muestran en los casos de uso.

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. Antes de describir los casos de uso, se muestra en la figura 4.1 el diagrama de casos de uso.

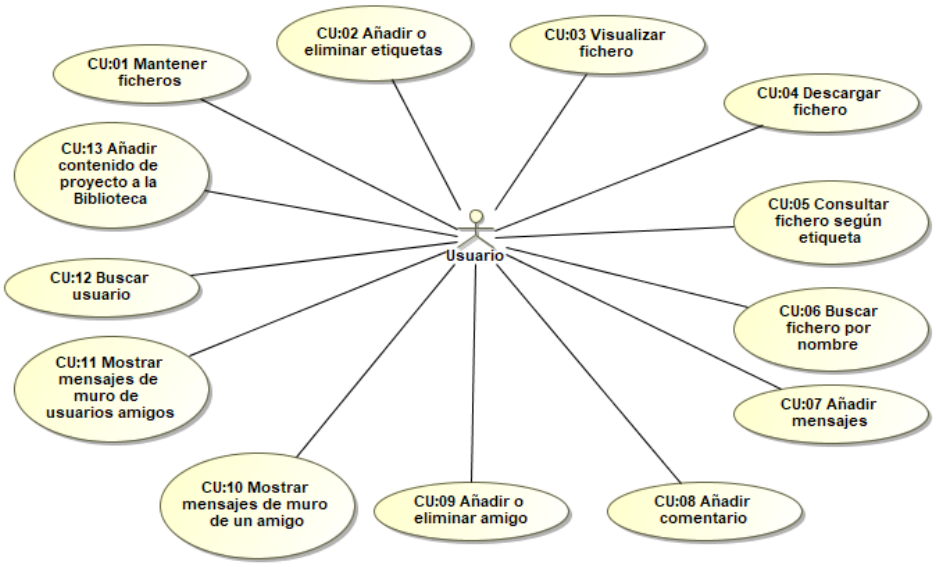

Figura 4.1: *Diagrama de casos de uso del proyecto realizado*

Cuadro 4.1: **CU.01 Mantener ficheros**

Especificación del caso de uso	
Identificador	CU01
Nombre	Mantener ficheros
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite al usuario subir, eliminar o modificar datos de los ficheros.
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU03, CU04, CU05, CU06
Historias de usuario relacionadas	HU-02, HU-03, HU-04, HU-09 y HU-15
Precondición	Para llevar a cabo una modificación de los datos o un borrado debe primero existir el fichero.
Trigger	El usuario necesita añadir, modificar o subir un fichero a Edueca.
Secuencia normal	**Acción**
1	El usuario selecciona la opción subir/Modificar/Borrar fichero
2	El sistema permite al usuario seleccionar un fichero
3	El usuario introduce la información del producto (en el caso de subir o modificar)
4	El sistema añade, elimina o modifica un fichero. Una vez realizado esto actualiza el tamaño total consumido por los ficheros del usuario
Importancia	Alta
Prioridad	corto plazo
Comentarios	N/A

Cuadro 4.2: **CU.02 Añadir o eliminar etiquetas**

Especificación del caso de uso	
Identificador	CU02
Nombre	Añadir o eliminar etiquetas
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite al usuario añadir o eliminar etiquetas sobre un fichero.
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU05
Historias de usuario relacionadas	HU-05, HU-07 y HU-10
Precondición	Debe existir un fichero para poder insertar etiquetas
Trigger	El usuario necesita añadir o quitar etiquetas de un fichero.
Secuencia normal	
1	El usuario selcciona eliminar/añadir etiquetas
2	El sistema permite seleccionar un fichero
3	El usuario introduce el nombre de la etiqueta/etiquetas
4	El sistema eliminar o añade las etiquetas
Excepciones	
4	Si el fichero no tiene etiquetas no se pueden borrar.
3	Si un fichero ya contiene una etiqueta el sistema mostrara un mensaje de error.
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.3: **CU.03 Visualizar fichero**

Especificación del caso de uso	
Identificador	CU03
Nombre	Visualizar fichero
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	Cuando un fichero sea una imagen, un pdf o un formato de texto el sistema debe permitir visualizar su contenido
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU01,CU06
Historias de usuario relacionadas	HU-21
Precondición	El fichero debe tener el formato adecuado (pdf, imagen o documento de texto)
Trigger	El usuario quiere ver el contenido de un fichero
Secuencia normal	
1	El usuario selecciona visualizar fichero
2	El sistema permite seleccionar un fichero
3	EL usuario selecciona un fichero
4	El sistema muestra el contenido del fichero con la aplicación adecuada.
Excepciones	
3	Si el fichero elegido no tiene el formato adecuado el sistema no mostrará una visualización del mismo
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.4: **CU.04 Descargar fichero**

Especificación del caso de uso	
Identificador	CU04
Nombre	Descargar fichero
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite descargar el fichero seleccionado por el usuario
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU01
Historias de usuario relacionadas	HU-06
Precondición	Deben existir ficheros en el sistema de ese usuario para poder descargarlo
Trigger	El usuario quiere obtener un fichero del sistema
Secuencia normal	
1	El usuario selecciona descargar fichero
2	El sistema muestra los fichero del usuario
3	El usuario selecciona el fichero
4	El sistema envia el fichero al usuario
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.5: **CU.05 Consultar fichero según etiqueta**

Especificación del caso de uso	
Identificador	CU05
Nombre	Consultar fichero según etiqueta
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite al usuario buscar ficheros según las etiquetas que tengan asignadas.
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU01,CU02
Historias de usuario relacionadas	HU-08
Precondición	Algún fichero del usuario debe tener etiquetas
Trigger	El usuario quiere buscar los ficheros que contienen una o varias etiquetas
Secuencia normal	
1	El usuario selecciona buscar fichero por etiquetas
2	El sistema muestra las etiquetas existentes en los ficheros del usuario
3	El usuario selecciona las etiquetas deseadas
4	El sistema muestra los ficheros que coinciden con la selección del usuario
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.6: **CU.06 Buscar fichero por nombre**

Especificación del caso de uso	
Identificador	CU06
Nombre	Buscar fichero por nombre
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite al usuario buscar un fichero según el nombre del fichero
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU01
Precondición	Debe existir algún fichero de ese usuario en el sistema
Trigger	El usuario quiere buscar un fichero según el nombre del fichero.
Secuencia normal	
1	El usuario selecciona buscar fichero
2	El sistema permite introducir el nombre del fichero a buscar
3	EL usuario introduce el nombre del fichero
4	El sistema muestra el fichero o ficheros que se ajusten a la busqueda
Excepciones	
3	No hay ningún fichero con ese nombre, el sistema muestra un mensaje de fichero no encontrado
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.7: **CU.07 Añadir mensaje**

Especificación del caso de uso	
Identificador	CU07
Nombre	Añadir mensaje
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite escribir un mensaje en el apartado de red en el muro personal
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU10, CU13
Precondición	N/A
Trigger	El usuario quiere añadir un mensaje en el muro personal
Secuencia normal	
1	El usuario selecciona añadir mensaje
2	El sistema permite a al usuario escribir un mensaje
3	El usuario introduce el contenido del mensaje
4	El sistema añade el mensaje con el contenido introducido por el usuario
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.8: **CU.08 Añadir comentario**

Especificación del caso de uso	
Identificador	CU08
Nombre	Añadir comentario
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite al usuario añadir un comentario sobre un mensaje
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	N/A
Precondición	Tiene que existir un mensaje sobre el que hacer el comentario
Trigger	El usuario quiere añadir un comentario en un mensaje
Secuencia normal	
1	El usuario selecciona realizar comentario
2	El sistema permite seleccionar el mensaje sobre el hacer el comentario
3	El usuario introduce el comentario del mensaje
4	El sistema introduce el comentario
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.9: **CU.09 Añadir o eliminar amigo**

Especificación del caso de uso	
Identificador	CU09
Nombre	Añadir o eliminar amigo
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite añadir un usuario como amigo con el objetivo de que después el usuario pueda ver los mensajes de sus amigos. El sistema permite eliminar un usuario como amigo para no ver sus mensajes
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	N/A
Historias de usuario relacionadas	HU-19 y HU-20
Precondición	N/A
Trigger	El usuario quiere añadir o eliminar a un usuario como amigo
Secuencia normal <Añadir>	
1	El usuario selecciona añadir un usuario como amigo
2	El sistema permite buscar un usuario según el nombre
3	El usuario introduce el nombre a buscar
4	El sistema muestra los usuarios que coinciden con la búsqueda
5	El usuario selecciona el usuario que quiere añadir como amigo
6	El sistema añade ese usuario como amigo
Secuencia normal <Eliminar>	
1	El usuario selecciona añadir un usuario como amigo
2	El sistema permite buscar un usuario según el nombre
3	El usuario introduce el nombre a buscar
4	El sistema muestra los usuarios que coinciden con la búsqueda
5	El usuario selecciona el usuario que quiere eliminar como amigo
6	El sistema quita ese usuario como amigo
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.10: **CU.10 Mostrar mensajes de muro de usuario**

Especificación del caso de uso	
Identificador	CU10
Nombre	Mostrar muro de usuario
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema muestra los mensajes de un usuario seleccionado
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU07, CU09
Historias de usuario relacionadas	HU-17
Precondición	N/A
Trigger	El usuario quiere ver muro de un usuario
Secuencia normal	
1	El usuario selecciona ver muro de usuario
2	El sistema permite buscar un usuario
3	El usuario selecciona el usuario que quiere
4	El sistema muestra el muro con los mensajes del usuario seleccionado
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.11: **CU.11 Mostrar mensajes de amigos**

Especificación del caso de uso	
Identificador	CU11
Nombre	Mostrar todos los mensajes de amigos
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema muestra los mensajes de los usuarios que son amigos
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	CU07, CU09
Historias de usuario relacionadas	HU-17
Precondición	N/A
Trigger	El usuario quiere ver mensajes de amigos
Secuencia normal	
1	El usuario selecciona ver mensajes de amigos
2	El sistema muestra los mensajes de todos los usuarios amigos
Excepciones	
2	Si el usuario no tiene amigos añadidos entonces el sistema muestra un mensaje de no hay mensajes
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.12: **CU.12 Buscar usuario**

Especificación del caso de uso	
Identificador	CU12
Nombre	Buscar usuario
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite buscar un usuario por el nombre
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	N/A
Historias de usuario relacionadas	HU-18
Precondición	N/A
Trigger	El usuario quiere buscar a un usuario
Secuencia normal	
1	El usuario selecciona buscar un usuario
2	El sistema permite introducir el nombre del usuario a buscar
3	El usuario introduce el nombre a buscar
4	El sistema muestra usuarios que coinciden con la busqueda
Excepciones	
3	Si el usuario introduce un nombre que no coincide con ningún nombre de usuario el sistema muestra un mensaje de no encontrado
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

Cuadro 4.13: **CU.13 Añadir contenido del proyecto a la biblioteca**

Especificación del caso de uso	
Identificador	CU13
Nombre	Añadir contenido del proyecto a la biblioteca
Versión	1.0
Autores	Javier
Fuentes	Product Owner
Descripción	El sistema permite añadir un fichero nuevo a partir del contenido de un curso
Nivel	Tarea principal
Actor principal	Usuario
Actores secundarios	N/A
Relaciones	N/A
Historias de usuario relacionadas	HU-13 y HU-15
Precondición	Debe existir contenido en el curso
Trigger	El usuario quiere guardar un contenido de un curso en la biblioteca
Secuencia normal	
1	El usuario selecciona añadir contenido a la biblioteca
2	El sistema genera un fichero con el contenido
3	El sistema añade el fichero a la biblioteca
4	El sistema actualiza y muestra el espacio disponible
Importancia	Alta
Prioridad	Corto plazo
Comentarios	N/A

4.1.2. Requisitos no funcionales

Se trata de una característica requerida del sistema, del proceso de desarrollo, del servicio prestado o de cualquier otro aspecto del desarrollo, que señala una restricción del mismo. No describen información a guardar, ni funciones a realizar, sino características de funcionamiento, por eso suelen denominarse Atributos de calidad de un sistema.

Cuadro 4.14: **RNF01 Interfaz**

Identificador	RNF01
Nombre	Interfaz
Descripción	La interfaz debe ser fácil, intuitiva y amable.
Relaciones	HU01, HU14, HU16, HU17
Comentarios	La gran mayoría de usuarios finales son personas mayores y la interfaz debe adecuarse a sus necesidades para que la usen con naturalidad

Cuadro 4.15: **RNF02 Respuesta**

Identificador	RNF02
Nombre	Respuesta
Descripción	El tiempo de respuesta de las distintas acciones no debe ser mayor de 1 o 2 segundos para evitar confusiones y malas experiencias de uso. Un tiempo mayor que 1 o 2 segundos sin un mensaje de información al usuario puede dar la sensación de fallo.

4.1.3. Requisitos de datos

Se trata de la información que debe mantener el sistema, define los objetos y los atributos de los mismos.

Cuadro 4.16: **RD01 Ficheros**

Identificador	RD01
Nombre	Ficheros
Datos	Id, Datos, tipo, tamaño, nombre, fecha de creación, fecha de borrado
Relaciones	RD02
Comentarios	Datos será toda la información que conforma el fichero tal y como es.

Cuadro 4.17: **RD02 Etiquetas**

Identificador	RD02
Nombre	Etiquetas
Datos	Identificador, nombre de la etiqueta
Relaciones	RD01

Cuadro 4.18: **RD03 Mensajes**

Identificador	RD03
Nombre	Mensajes
Datos	Id, texto del mensaje, fecha de creación, fecha de modificación, fecha de borrado
Relaciones	RD04, RD05

Cuadro 4.19: **RD04 Comentarios**

Identificador	RD04
Nombre	Comentarios
Datos	Id, texto del mensaje, fecha de creación, fecha de borrado
Relaciones	RD03

Cuadro 4.20: **RD05 Grupos**

Identificador	RD05
Nombre	Grupos
Datos	Id, Nombre , fecha de creación, fecha de borrado
Relaciones	RD03

Cuadro 4.21: **RD06 Usuarios**

Identificador	RD06
Nombre	Usuarios
Datos	Id, Nombre, Apellido , contraseña, email, alias, telefono, imagen , fecha de creación, fecha de borrado , Rol
Relaciones	RD03

4.2. Diseño del sistema

En este apartado se desarrollan las directrices definidas durante el análisis y se presenta el diseño del sistema que se desarrollará durante la estancia en prácticas.

4.2.1. Diagrama de clases

Sirve para visualizar las relaciones entre las clases que involucran el sistema y la forma en la que se relacionan entre ellas. La figura 4.2 muestra el diagrama de clases de este proyecto.

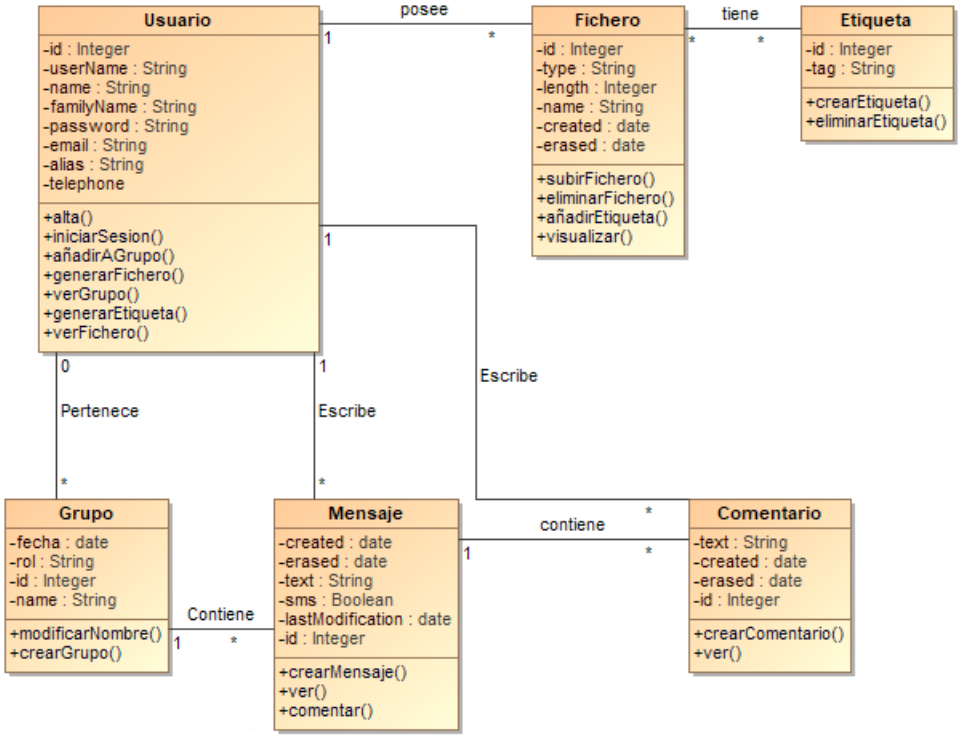

Figura 4.2: *Diagrama de clases*

Ahora se va desarrollar una explicación sobre las distintas clases que aparecen en el diagrama:

- Fichero: esta clase representa todos los archivos que guardan los usuarios en la biblioteca.
- Etiqueta: la clase Etiqueta representa los distintos nombres que se pueden asignar a los ficheros para agruparlos o diferenciarlos unos de otros. Las etiquetas se introducen en la plataforma como un método organizativo para el usuario.
- Usuario: esta clase identifica a todos los usuarios de que existen en Edueca. Tiene relación con fichero por los ficheros que posee un usuario en la biblioteca y además tiene otra relación con fichero por la imagen que selecciona el usuario como imagen de perfil.

- Grupo: esta clase representa las distintas asociaciones de usuarios que se generan en el apartado de red con motivo de compartir mensajes entre ellos. Esta clase se relaciona con mensaje ya que los mensajes son el contenido de los grupos en la red.

- Mensaje: esta clase representa los distintos textos que se publican en el apartado de red, son textos creados por el usuario y están asociados con un grupo de la red.

- Comentario: esta clase representa los distintos textos que se envían como respuesta a un mensaje, son creados por el usuario y se relacionan con un mensaje.

4.2.2. Diagrama de actividades

Muestra el flujo de actividades para los distintos casos de uso de la Biblioteca y la Red a través de una serie de acciones, representa las acciones que un usuario puede realizar que se han generado a partir de este proyecto.

En la figura 4.3 se muestra primeramente que el usuario elige entre acceder al apartado de biblioteca o al de red, y una vez tomada la decisión se muestran todas las acciones que puede llevar a cabo en cada uno de los apartados.

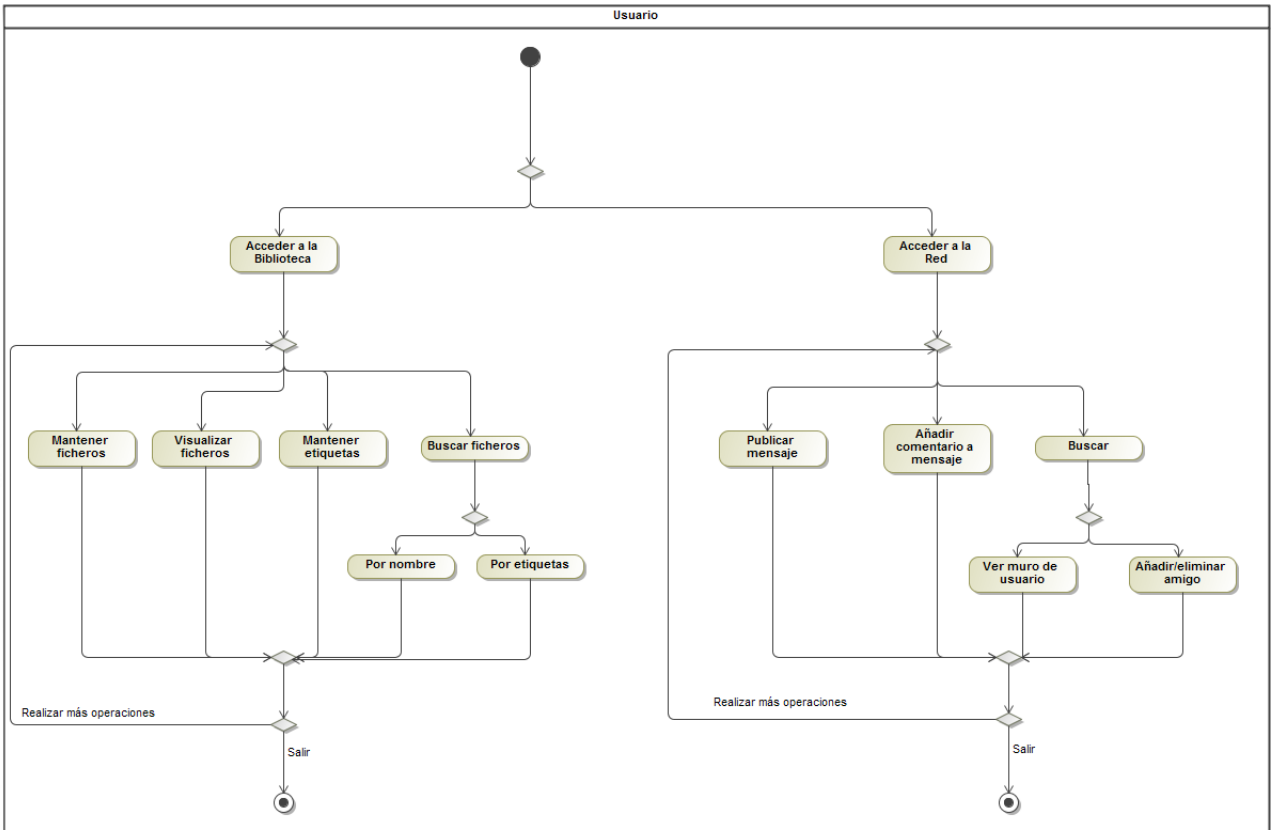

Figura 4.3: *Diagrama de actividades*

4.3. Diseño de la interfaz

En este apartado se muestran los 5 mockups que se han creado para definir la apariencia aproximada que deberán tener los distintos apartados generados durante la estancia en prácticas.

4.3.1. Página de biblioteca

El mockup de la figura 4.4 se ajusta a la página donde se muestran los ficheros del usuario, esta página será llamada "Biblioteca". En dicha página se permitirá elegir ficheros para borrarlos o para ver la información del mismo en el menú de la derecha, cambiar el nombre, añadir/eliminar etiquetas y buscar ficheros por nombre.

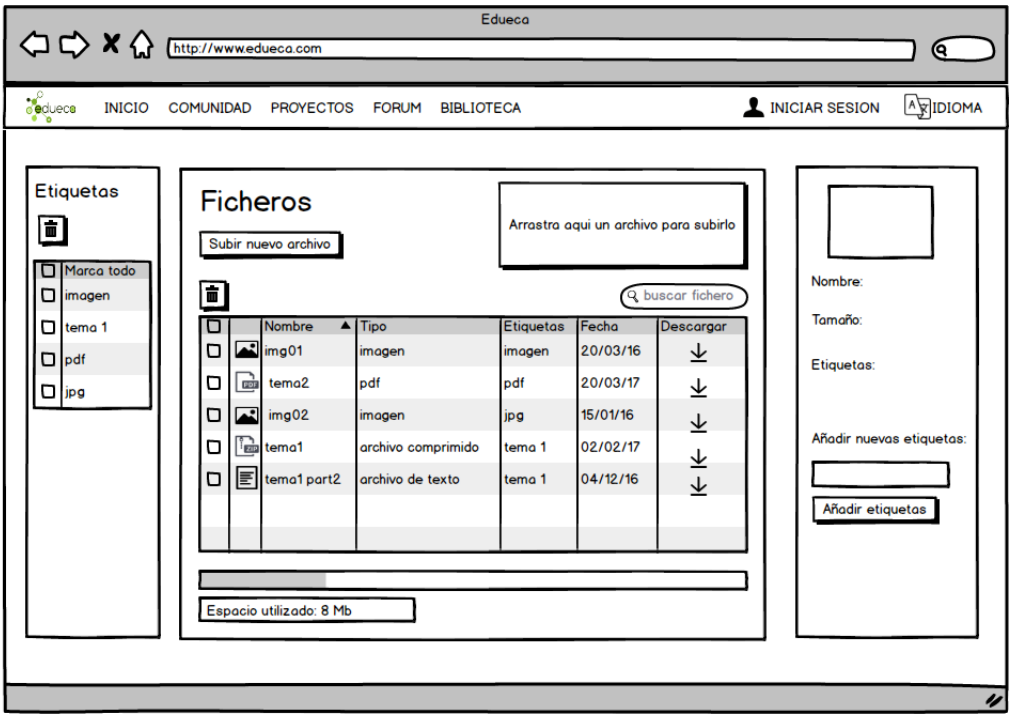

Figura 4.4: *Mockup página de biblioteca*

En el lateral izquierdo se muestra el menú que permite elegir etiquetas para que solo se muestren los ficheros que contengan las etiquetas seleccionadas en el menú. Este selección también sirve para eliminar las etiquetas de los ficheros en el caso de seleccionar el botón que tiene una papelera.

4.3.2. Subir nuevo fichero

El mockup de la figura 4.5 es de la página donde el sistema permite subir un nuevo fichero, renombrarlo con el nombre deseado y añadirle las etiquetas deseadas. A esta página se accede pulsando al botón "Subir nuevo archivo"que se muestra en la figura 4.4.

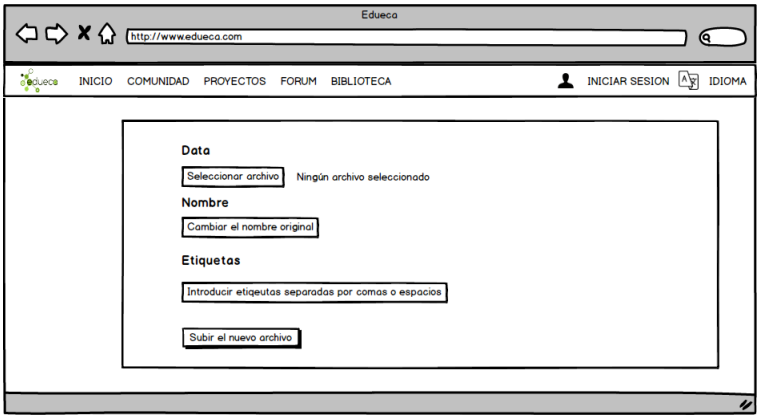

Figura 4.5: *Mockup subir nuevo fichero*

4.3.3. Página de red

El mockup de la figura 4.6 muestra los mensajes, y los comentarios que se han hecho sobre ellos, publicados por aquellos usuarios que el usuario ha añadido previamente como amigo.

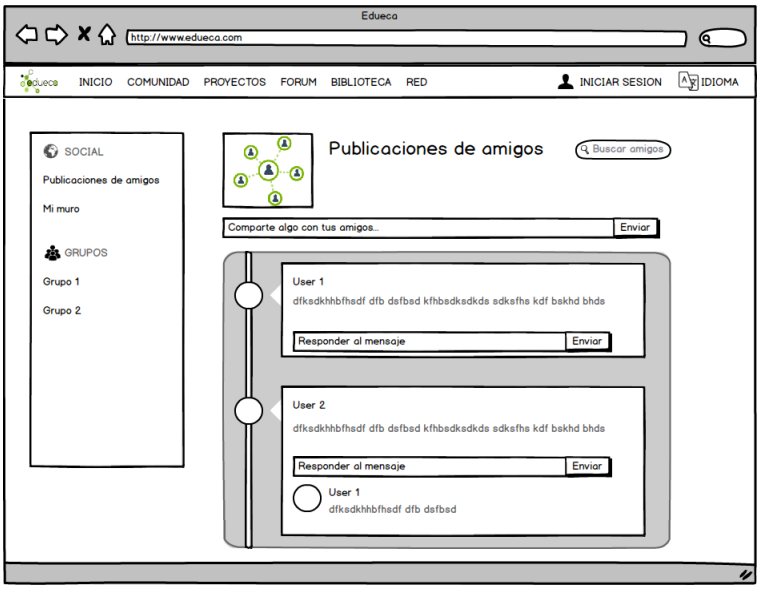

Figura 4.6: *Mockup apartado de red*

4.3.4. Buscar usuario

El mockup de la figura 4.7 muestra la pantalla que se genera tras introducir un nombre en el buscador que se muestra en la figura anterior.

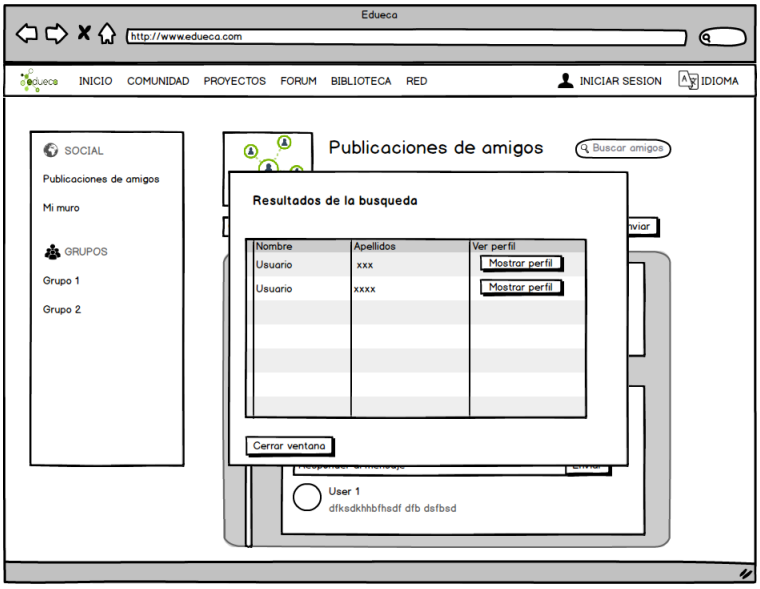

Figura 4.7: *Mockup buscar usuario*

4.3.5. Muro personal

El mockup de la figura 4.8 muestra la pantalla en la que se muestran solo los mensajes publicados por el usuario.

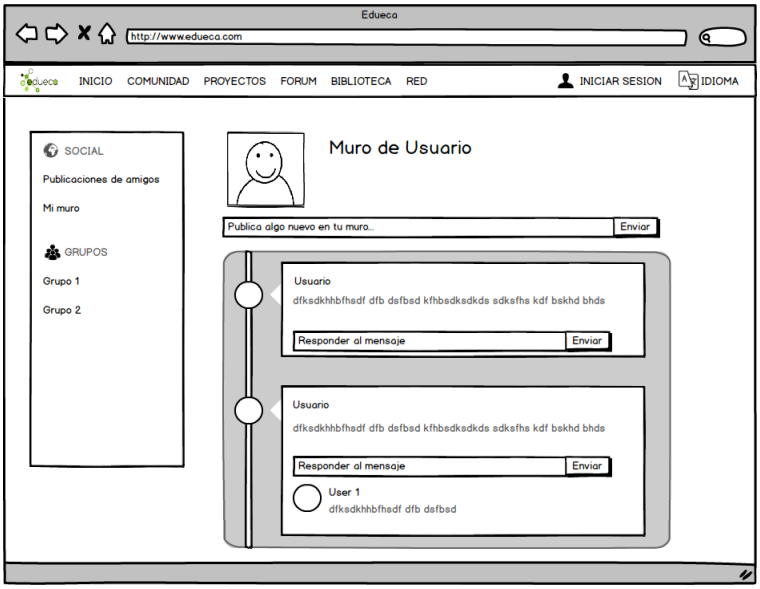

Figura 4.8: *Mockup muro personal del apartado de red*

Capítulo 5

Implementación y pruebas

Toda la implementación de Edueca está desarrollada en Symfony por lo que todo el proyecto desarrollado en la estancia en prácticas también se desarrolla en Symfony. Symfony se basa en el modelo Vista Controlador (figura 5.1).

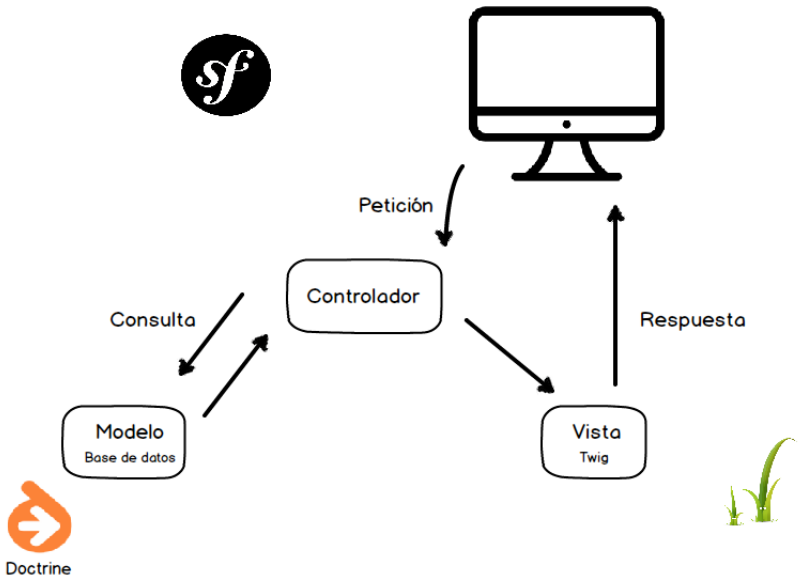

Figura 5.1: *Modelo vista controlador de Symfony*

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones. El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.

El código de Symfony se basa en Bundles. Los Bundles son un conjunto estructurado de archivos que se encuentran en un directorio y que implementan una característica concreta.

Para este proyecto se han utilizado dos Bundles: el AppBundle, que es un bundle que ya existía en la plataforma, que se ha editado para introducir las entidades y el controlador de la Biblioteca; y un Bundle que se ha generado nuevo llamado NetBundle donde se encuentra el controlador del apartado de red y las entidades que lo forman.

Se va a mostrar la implementación de las distintas historias de usuario según se desarrollaron en los distintos Sprints.

5.1. Sprint 1

En la reunión para el primer sprint se habían planeado las siguientes historias de usuario: HU-01, HU-02, HU-08 y HU-14. Que son:

- **HU-01 Pantalla para visualizar ficheros**

  Realizar un apartado para que el usuario pueda ver sus ficheros. *Story point: 5*

- **HU-02 Subir ficheros**

  El usuario pueda subir un fichero especificando un nombre nuevo y también puede añadir etiquetas. *Story point: 8*

- **HU-08 Buscar ficheros por etiquetas**

  El usuario pueda buscar ficheros por etiquetas para agilizar la navegación entre sus ficheros. *Story point: 3*

- **HU-14 Redirigir a la pantalla de login**

  Como programador quiero que si un usuario no esta identificado al entrar en la biblioteca entonces se le redirige a la pantalla de identificación o registro. *Story point: 2*

5.1.1. Desarrollo del sprint

Se comenzó realizando primeramente un controlador para las acciones que tuvieran que realizarse en el apartado de la biblioteca, para ello se generó el fichero libraryController.php y se añadió al Bundle AppBundle, una vez realizado esto, y dado que la entidad fichero ya existía en el sistema, se generó la entidad Tag que representa a las etiquetas que se pueden añadir a los ficheros. Una vez generado esta entidad se crearon dos plantillas *Twig*:

- Una que mostraría un formulario para poder subir un fichero, ponerle un nombre (o dejarlo con el nombre original del fichero) y poder añadirle etiquetas al fichero que se sube.
- Y la segunda que sería para poder visualizar todo el apartado de biblioteca.

Una vez generadas las plantillas twig se añadieron dos funciones al controlador para cuando se llamara a las mismas se mostrara cada plantilla *Twig*.

Más tarde se creo un formulario de Symfony en relación a la entidad fichero para poder subir mediante el formulario un fichero nuevo a la plataforma, además en la plantilla *Twig* se introdujo un Select2 B.1 para poder introducir las etiquetas que se quieren añadir al fichero.

5.1.2. Problemas surgidos

Debido a la falta de experiencia con el IDE PhpStorm y con Symfony, el avance en este Sprint no fue tan rápido como se podría esperar al inicio. Por ello no se realizaron todas las historias de usuario planificadas para este Sprint y solo se realizó la HU-02. Por tanto, el resto de historias restantes se realizaron en el segundo sprint.

Los problemas surgidos en este sprint se centraron en el formulario para subir un fichero a la plataforma debido a que los formularios en Twig son algo complejos y además se pretendía añadir un select2 junto al formulario. La complejidad radicaba en que el select2 no forma parte del formulario de Twig por defecto por lo que cuando se hacía submit en el formulario había que realizar una llamada AJAX para generar objetos de la entidad etiquetas a partir de los strings introducidos en el Select2, para que una vez el formulario realizara el submit de verdad entonces se le pasaran objetos de esa entidad y no string al formulario de Twig.

5.1.3. Burn down chart

Para ver como se realizan las historias de usuario en comparación con la estimación ideal se mostrará el burn down chart tras cada sprint. En la figura 5.2 se muestra el gráfico tras el primer sprint. Se observa que no se ajusta a la previsión ideal, sin embargo esto se achaca a la falta de experiencia y se espera un incremento de velocidad en los siguientes sprints.

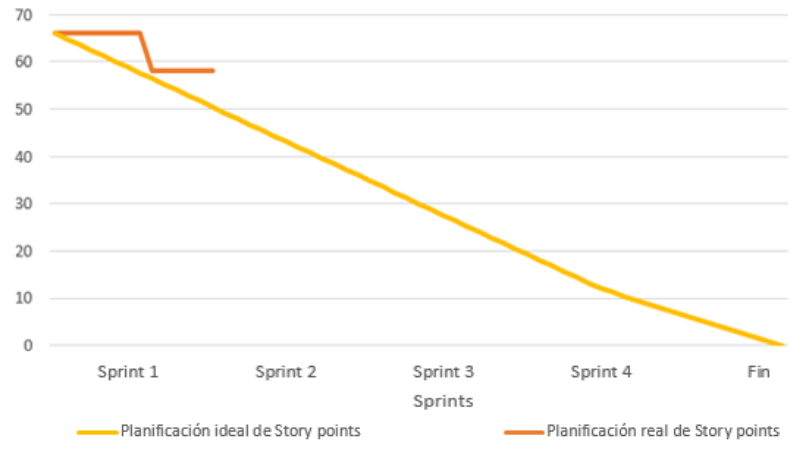

Figura 5.2: *Burn down chart tras el primer sprint*

5.2. Sprint 2

En la reunión previa al segundo sprint se planificaron las historias de usuario que se llevarían a cabo en este sprint, considerando que las historias de usuario que no se finalizaron en el sprint anterior se completarían en este junto al resto de tareas ya que el alumno ya contaría con la experiencia suficiente como para realizar las historias de usuario de forma más veloz. Las historias de usuario planificadas para este sprint fueron: HU03, HU04, HU05, HU06, HU07, HU09 y HU11. Que se corresponden con:

- **HU-03 Borrar ficheros**

  El usuario pueda borrar sus archivos. *Story point: 2*

- **HU-04 Subir ficheros arrastrándolos**

  Como programador quiero añadir un *dropzone* B.2 en la zona de los archivos para poder arrastrar un archivo y que se suba a la plataforma. *Story point: 2*

- **HU-05 Gestionar etiquetas en los ficheros**

  El usuario pueda añadir y quitar etiquetas de un archivo para poder filtrar los archivos por etiquetas. *Story point: 3*

- **HU-06 Descargar de la biblioteca**

  El usuario pueda descargar sus archivos desde la biblioteca. *Story point: 2*

- **HU-07 Añadir etiqueta a muchos ficheros**

  El usuario pueda añadir la misma etiqueta a los ficheros seleccionados para agilizar el añadir la misma etiqueta a muchos archivos. *Story point: 2*

- **HU-09 Cambiar nombre de fichero**

  El usuario pueda cambiar el nombre de un fichero. *Story point: 2*

- **HU-11 Ver imagen**

  El usuario pueda visualizar la imagen si al seleccionar un fichero es una imagen . *Story point: 5*

5.2.1. Desarrollo del sprint

Se comenzó el sprint realizando una pantalla para mostrar los ficheros de los usuarios (HU01) para ello se creo un función nueva en el controlador que realizaba una consulta a la base de datos que devolvía aquellos ficheros de los que fuera propietario el usuario. Una vez obtenidos los ficheros se mostraba la plantilla generada para ello en el sprint anterior y se cargaban los datos de los ficheros en una tabla de *Bootstrap*.

La siguiente historia de usuario que se llevo a cabo fue la HU03 que permite borrar un fichero, esta función se realizó utilizando *Bootstrap* ya que permite seleccionar filas de la tabla y contiene una función en *Javascript* que devuelve los elementos seleccionados. Una vez obtenidos

los elementos seleccionados se realiza una llamada a una función que modifica en la tabla de ficheros el atributo borrado y lo pasa de null a la fecha de borrado. Esto se hace así debido a que los ficheros de la plataforma no se pueden borrar y deben de mantenerse por motivos legales. Sin embargo, una vez se cambia el valor del atributo borrado y no está a nulo al usuario no se le muestra el fichero. Además, aquellos ficheros "borrados"no se tienen en cuenta para calcular el espacio consumido por ese usuario.

Más adelante se realizó la historia de usuario HU04. Para ello se agregó un *dropzone* que permite arrastrar ficheros encima del contenedor del *dropzone* y que se suban a la plataforma. De esta forma se consigue subir ficheros de forma rápida a la plataforma. Sin embargo, no es posible añadirles etiquetas antes de subirlos.

Para la historia de usuario HU05 (Gestionar etiquetas en los ficheros), se utilizó *Bootstrap* para permitir al usuario seleccionar los ficheros deseados. Una vez se obtiene la selección del usuario se muestra una lista de las etiquetas del fichero con un botón para borrarlas si lo desea y un *select2* para permitir introducir nuevas etiquetas de la misma forma que se hacía en la pantalla de subir nuevo fichero en el sprint 1. Para estas acciones se crearon las pertinentes funciones en el controlador. En el caso de borrar una etiqueta se mandaba al controlador los identificadores de la etiqueta y del fichero, de forma que se borraba la fila de la base de datos que relaciona la etiqueta realizando una consulta con estos dos identificadores. Respecto a añadir nuevas etiquetas al fichero, la función creada itera sobre el vector de strings que se le manda con las etiquetas, genera una etiqueta en el caso de que no exista ya una con ese string y crea una relación con el fichero introduciendo una nueva fila en la tabla correspondiente en la base de datos.

La siguiente historia de usuario que se llevó a cabo fue la HU06 (Descargar de la biblioteca). Esta funcionalidad se llevo a cabo introduciendo un botón en cada fila de la tabla de *Bootstrap*, este botón contiene una referencia a la URL del archivo, esta URL se introduce al cargar la tabla con los ficheros, y una vez pulsas sobre el botón se descarga el contenido de la URL al equipo del usuario.

En el sprint anterior no se terminó la historia de usuario HU08 (Buscar ficheros por etiquetas), por lo que se realizó en el sprint 2. Para llevar a cabo esta historia de usuario se introdujo una segunda tabla *bootstrap* en la página que contiene todas las etiquetas de los ficheros de ese usuario, de forma que según el usuario elija las etiquetas en dicha tabla se hace una llamada *AJAX* a una nueva función del controlador que devuelve solo los ficheros que contienen alguna de las etiquetas marcadas por el usuario en la tabla. Una vez termina la función, la respuesta se carga en la tabla *Bootstrap* central de los ficheros para que el usuario ver los ficheros con dichas etiquetas.

Para la historia de usuario HU09 (Cambiar nombre de fichero), primero se obtiene el fichero seleccionado y desde el *Javascript* se comprueba que solo hay un fichero seleccionado. Una vez se comprueba, se realiza una llamada *AJAX* a una función del controlador de biblioteca que recoge el nuevo nombre y el identificador del fichero de la llamada *AJAX* y con esos datos hace un *update* sobre la tabla fichero y se pone el nuevo nombre del fichero.

La historia de usuario HU14 (Redirigir a la pantalla de login) tampoco se realizó durante le primer sprint. Esta historia de usuario pretende que si un usuario que no ha iniciado sesión pretende entrar en la página entonces se muestre una pantalla para que se registre en la pla-

taforma o para que inicie sesión. Para ello se introdujo una condición en la plantilla *Twig* de forma que si la variable de usuario estaba a 'anon.' se redirige a la plantilla *Twig* de inicio de sesión.

Para realizar la historia de usuario HU11 se añadieron ventanas modales mediante *Javascript* de forma que una vez el usuario seleccionara un fichero en el caso de que sea una imagen se mostrara una ventana modal con la visualización de la imagen. Para ello se realizaba una llamada *AJAX* que obtiene un thumbnail de la imagen del tamaño deseado y dicho thumbnail se introduce en la ventana modal.

5.2.2. Problemas surgidos

El problema principal de este sprint radicaba en que para obtener los ficheros seleccionados en la table *Bootstrap* debía actualizar el *Bootstrap* de toda la plataforma, con lo que antes de actualizarlo primero era necesario revisar todo el proyecto para comprobar que la actualización causaba algún error en el resto de la plataforma.

5.2.3. Burn down chart

Durante este Sprint se realizaron las historias de usuario que se quedaron por hacer en el sprint anterior y las historias de usuario previstas para el segundo sprint, esto se debe a que tras el primer sprint el alumno ya conoce el funcionamiento de Symfony lo suficiente como para avanzar con rapidez. Además debido al uso de *Javascript* en la mayoría de las historias de usuario (que ha sido utilizado en algunas asignaturas de la carrera) era sencillo avanzar de forma más veloz.

Tras la finalización del segundo Sprint el desarrollo real de los story points se ajusta mucho más al desarrollo ideal y el burn down chart resultante es el que se muestra en la siguiente figura 5.3.

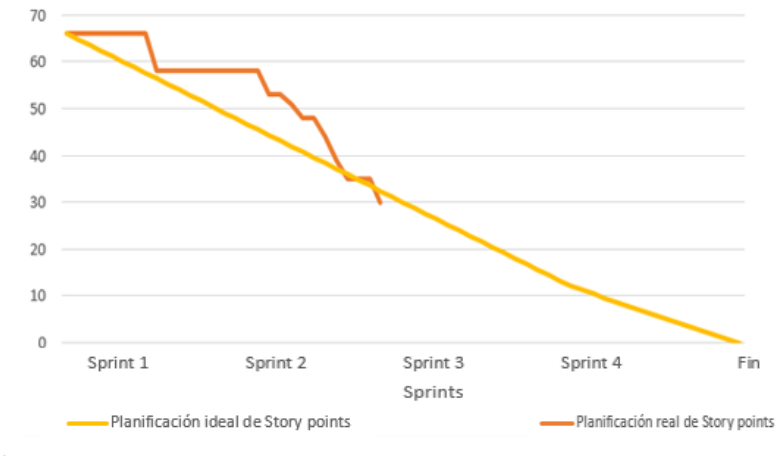

Figura 5.3: *Burn down chart tras el segundo sprint*

5.3. Sprint 3

Tras los dos primeros sprints, las historias de usuario asignadas al tercer sprint fueron: HU10, HU12 y HU13. Estas historias de usuario corresponden con:

- **HU-10 Borrar una etiqueta de todos los ficheros**

  El usuario pueda borrar una etiqueta en el menú de etiquetas para que se borre de todos los ficheros de ese usuario. *Story point: 3*

- **HU-12 Añadir fichero de la biblioteca a un contenido de un curso o proyecto**

  Como profesor o creador de un proyecto quiero poder seleccionar un fichero de la biblioteca para añadirlo a un contenido que se está creando en un proyecto o curso. *Story point: 5*

- **HU-13 Generar PDF a partir del contenido**

  El usuario pueda añadir a la biblioteca un PDF generado a partir de un contenido o un texto que se encuentra en un proyecto. *Story point: 8*

5.3.1. Desarrollo de sprint

Se comenzó el sprint desarrollando la historia de usuario HU13. Esta historia de usuario era compleja debido a que el objetivo de la misma consistía en convertir un trozo de código en *HTML* en un pdf que se añadiría posteriormente a la biblioteca. Para realizar esta historia de usuario primero hubo que realizar un búsqueda entre las librerías disponibles en *C*, *Python*, *Java*, *Javascript* y *PHP* para encontrar una que realizara la conversión del código *HTML* en *PDF*. Sin embargo después de probar las distintas opciones se optó por buscar un Bundle de Symfony que realizara dicha tarea. El Bundle elegido fue TCPDF. Este Bundle se puede añadir al proyecto utilizando el Composer B.3.

Una vez instalado el bundle hubo que implementar las clase que genera el Bundle y sobreescribir los métodos de cabecera y pie de páginas del mismo, con el objetivo de editarlos y que la cabecera mostrara información del proyecto del que procede el contenido y en el pie de página información legal de Edueca. Tras sobreescribir los métodos se generó una función en el controlador. Esta función se encargaba de generar un objeto de la clase mencionada anteriormente. Además debido a opciones de seguridad de los servidores y los navegadores no es posible obtener las imágenes a partir del *HTML* en algunos casos. Por ello, durante la ejecución que generaba el PDF era necesario buscar las imágenes del código, obtener el id de las mismas y realizar una consulta a la base de datos. Una vez obtenida la imagen se transformaba a *base64* [8] para que el Bundle pudiera introducirla correctamente en el PDF. Además de obtener la imagen de la base de datos, es necesario controlar el tamaño de la misma. Si la imagen fuera de gran tamaño, sería preciso ajustar su ancho, para que no fuera mayor que el ancho de un A4. Tras generar el PDF, se nombraba el fichero igual que el título del contenido en el proyecto, se incluía en la biblioteca del usuario y se actualiza el tamaño consumido por los ficheros del usuario.

La siguiente historia de usuario que se llevó a cabo fue la de permitir a un profesor, o un usuario que crea un proyecto, añadir un fichero de la biblioteca a un contenido del proyecto. Para esta historia de usuario era necesario conocer primero el funcionamiento del *ckeditor* B.4, por ello fue necesario buscar documentación del *ckeditor* para entender cómo podía incluirse un fichero en el contenido.

Primero se editó el *ckeditor* para que cuando se quisiera añadir un fichero se mostrara un ventana modal que tuviera el contenido de la biblioteca. A continuación, era necesario poder seleccionar un fichero y devolver al *ckeditor* la URL del mismo con el objetivo de que la insertara en la posición del contenido deseada por el usuario. Este proceso se lleva a cabo mediante un botón, que solo se muestra cuando la biblioteca es llamada desde el *ckeditor*, que envía mediante una llamada AJAX la selección de los ficheros a una función del controlador. Y por último es el controlador el que manda al *ckeditor* la URL del fichero seleccionado.

La última historia de usuario del sprint (HU10) consiste en borrar una etiqueta de todos los ficheros del usuario que la contengan, para ello se creó una función en el controlador de biblioteca que realiza una consulta en primer lugar sobre la tabla etiquetas y obtiene las etiquetas que se quieren borrar. Una vez se tienen los objetos etiquetas que se pretenden borrar se borran de la tabla que relaciona los ficheros con las etiquetas aquellos ficheros que sean del usuario y contengan alguna de esas etiquetas.

5.3.2. Problemas surgidos

El problema principal de este sprint se produjo para encontrar, instalar y configurar correctamente el bundle TCPDF ya que dependiendo de la versión de Symfony la instalación era distinta y no era trivial averiguar la instalación que se debía seguir. Una vez instalado hubo que realizar nuevamente una gran búsqueda de información para utilizarlo y editarlo correctamente. Por último fue necesario trabajar con strings para controlar las distintas imágenes. Esta tarea de trabajar con strings suele ser difícil y pesada por lo que fue más largo de lo previsto.

Otro problema significativo del sprint fue tanto encontrar los ficheros para editar el *ckeditor* dentro del propia proyecto como encontrar una documentación correcta que fuera útil para poder modificar el *ckeditor*.

El resto de tareas que se realizaron a lo largo del sprint fueron mucho más sencillas de realizar.

5.3.3. Burn down chart

El desarrollo del tercer sprint se llevó a cabo según lo planificado en la reunión anterior al inicio del mismo (figura 5.4)

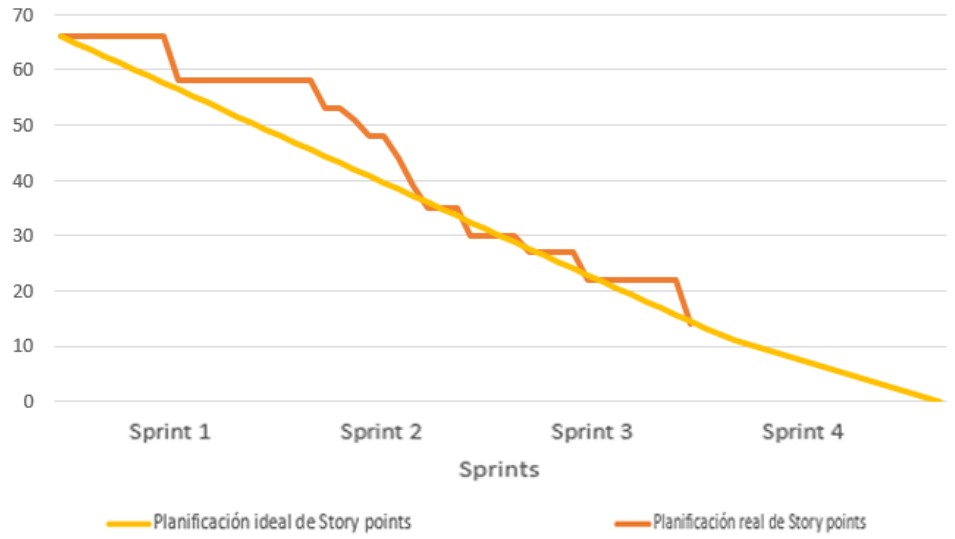

Figura 5.4: *Burn down chart tras el tercer sprint*

5.4. Sprint 4

Durante el sprint 4 se llevaron a cabo 2 historias de usuario, la HU15 y HU21, que se corresponden con:

- **HU-15 Mostrar tamaño consumido**

  Como programador quiero que se muestre en la parte baja de la biblioteca el tamaño total consumido en el sistema por los ficheros de ese usuario.*Story point: 3*

- **HU-21 Visualizar PDF o fichero de texto**

  El usuario pueda visualizar el contenido de un PDF o de un fichero de texto desde la biblioteca. *Story points: 8.*

Además se realizó el análisis para el desarrollo del apartado de red de la plataforma, se incorporaron las historias de usuario que corresponden con este apartado al backlog y se crearon las tablas necesarias en la base de datos.

5.4.1. Desarrollo de sprint

Para desarrollar la historia de usuario HU15 se modificó la función que suministra los datos de la biblioteca, se aprovechó la iteración que hace dicha función sobre los ficheros para obtener y sumar el tamaño de los ficheros del usuario. Una vez el controlador manda a la *plantilla Twig* A.3 que muestra la biblioteca los datos, se obtiene el tamaño total y se carga sobre una barra de proceso que muestra el porcentaje sobre 200MB que ha utilizado el usuario, además se muestra el tamaño total en MB.

La historia de usuario HU21 se realizó obteniendo la URL del fichero que se pretende visualizar mediante la selección suministrada por la tabla *Bootstrap*. Una vez se tiene la URL mediante una función *javascript*, se carga la URL en una ventana modal y se muestra la ventana modal al usuario. Debido a que los navegadores ya soportan mostrar un pdf o un fichero de texto no había que realizar más cambios ya que el propio explorador muestra sin problemas el contenido en la ventana modal.

Por último se pensaron y realizaron las tablas de usuario necesarias para el apartado de red. Eran necesarias las siguientes tablas:

- Mensajes: esta tabla contiene los mensajes escritos por los usuarios.
- Grupos: contiene los distintos grupos generados en la red, se considera un grupo tanto un muro de un usuario como un grupo generado por el mismo, pese a que estos últimos no se desarrollaran en la estancia en prácticas.
- Comentarios: contiene los mensajes que se escriben respondiendo a otros mensajes de la red.

Para realizar esta tarea en Symfony se generó un nuevo Bundle, NetBundle, con el objetivo de generar en el mismo las entidades y controladores necesarios para la red. Generando desde Symfony tres Entidades (mensajes, grupos y comentarios) después Doctrine genera las tablas necesarias en la base de datos.

5.4.2. Problemas surgidos

Los problemas de este sprint se basan en la generación del Bundle ya que era algo completamente nuevo, durante el proceso hubo problemas con las carpetas de cache que tuve que solucionar con ayuda de mi supervisor, ya que él sí tiene experiencia en la creación de Bundle.

5.4.3. Burn down chart

El siguiente gráfico muestra el burn down chart tras el cuarto sprint antes de introducir las nuevas historias de usuario referentes al apartado de red que se introducen antes del sprint 5. La figura 5.5 muestra la finalización de todas las historias de usuario del product backlog y cómo ha variado el desarrollo ideal de story points respecto al desarrollo real del mismo.

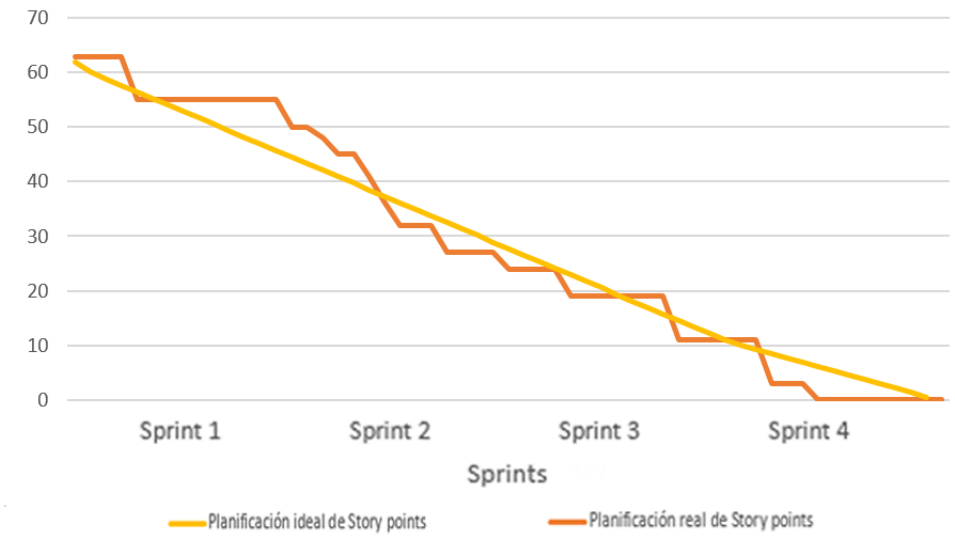

Figura 5.5: *Burn down chart tras el cuarto sprint*

5.5. Sprint 5

Para este último sprint había que desarrollar las historias de usuario: HU16, HU17, HU18, HU19 y HU20, estas historias de usuario fueron generadas en la reunión previa al sprint ya que durante los sprints anteriores se habían terminado las distintas historias de usuario del product backlog inicial. Las historias de usuario generadas eran:

- **HU-16 Pantalla de Muro**

  Un muro personal para cada usuario donde poder publicar mensajes y que otros usuarios puedan comentar a esos mensajes. *Story points:* 8.

- **HU-17 Mensajes de amigos**

  Un apartado donde visualizar los mensajes de los usuarios "amigos". *Story points:* 5.

- **HU-18 Buscar usuarios**

  Permitir a los usuarios buscar a otros usuarios. *Story points:* 8.

- **HU-19 Añadir a amigos**

  Permitir añadir un usuario como amigo con el fin de que sus publicaciones sean visualizadas en mi apartado de amigos. *Story points:* 3.

- **HU-20 Eliminar amigo**

  Permitir a un usuario eliminar un amigo. *Story points:* 3.

5.5.1. Desarrollo de sprint

Para desarrollar estas historias de usuario primero debían crearse las plantillas necesarias para mostrar el futuro contenido. Por ello se generaron dos plantillas *Twig*: una para mostrar el muro personal del usuario y otra para mostrar los mensajes de los usuarios añadidos como amigos.

Se comenzó el sprint con la historia de usuario HU16. Esta historia de usuario pretende que el usuario pueda publicar mensajes en un apartado personal que en un futuro el resto de usuarios puedan visitar. Para esta historia de usuario lo primero se generó una función en el controlador de red para mostrar la plantilla asignada, más tarde se introdujo un cuadro de texto con el que el usuario pudiera introducir sus mensajes y encima del cuadro una contenedor con la imagen de perfil del usuario. Una vez se permitía introducir mensajes se realizó el trabajo con el *CSS* para adaptar el diseño de la pantalla al prototipo generado anteriormente, además se modificó la función del controlador para que mandara los mensajes del usuario a la plantilla para poder mostrarlos más adelante.

Tras la historia de usuario HU16 se pasó a desarrollar la HU17 que muestra en una pantalla los mensajes publicados por usuarios amigos en sus muros o perfiles. Para desarrollar esta historia se usó parte del trabajo realizado para la historia anterior ya que el diseño de la pantalla es muy similar, el trabajo principal se desarrolló en la función del controlador creada para ello. En esta función se busca a los usuarios amigos y se obtienen los mensajes publicados por ellos en sus muros ordenados por fecha, y se mandan a la plantilla para cargar el contenido.

Una vez terminada la historia de usuario HU17 se pasó a desarrollar la HU18 que pretende poder buscar usuarios por nombre. Esta es la historia de usuario más compleja del sprint. En un principio intenté buscar un Bundle que realizara la búsqueda en la base de datos para obtener los usuarios que se ajustaran a la búsqueda, sin embargo ningún Bundle funcionaba suficientemente bien por lo que se buscó otra alternativa. La alternativa consiste en crear índices *fulltext* en la base de datos sobre los atributos nombre, apellidos, alias y email con el objetivo de que con esos indices la búsqueda devuelva los resultados que más se ajusten a la búsqueda.

Cuando la búsqueda de amigos funcionaba correctamente se realizaron las historias de usuario HU19 y HU20, ya que una vez un usuario podía buscar a otro entonces ya podía añadirlo o eliminarlo como amigo. Para desarrollar esta historia de usuario se introdujo un botón en los muros o perfiles de los usuarios buscados de manera que se permitiera añadirlo o eliminarlo dependiendo si el usuario ya era amigo o no. La forma de controlar si un usuario es amigo de otro es mirar si ese usuario pertenece al grupo que existe del muro del otro usuario (ya que el muro es un grupo en el que solo puede escribir su propietario), si ese usuario pertenece a ese grupo entonces el usuario propietario del muro lo ha añadido como amigo, en caso contrario no es amigo. Por tanto se generaron dos funciones que se llamaban desde los botones de añadir o eliminar amigo y que añadía al usuario buscado o lo eliminaba del grupo de muro del usuario.

Por último se generó una tercera plantilla para mostrar los muros o perfiles de otros usuarios, en esta tercera plantilla no se permite publicar mensajes, solo comentar sobre los mensajes que ha publicado dicho usuario. Además, se controló en la función que carga el muro una vez seleccionado el usuario desde la búsqueda que en el caso de que el usuario se buscara a sí mismo cargara la plantilla que muestra su muro personal.

Cabe destacar que en este sprint se desarrollaron mas story points que en los otros sprints anteriores. Esto se debe a que en la reunión previa al sprint se consideró que las historias de usuario tendrían más dificultad y por ello se asignaron muchos story points. Sin embargo, dentro de la primera historia de usuario se realizaron muchas tareas necesarias para el resto de historias de usuario y por tanto las historias de usuario fueron más sencillas de lo planificado.

5.5.2. Problemas surgidos

En este último sprint hubo dos problemas principales:

- El primero consistía en realizar el diseño correctamente de forma similar al prototipo planteado. En algunos casos trabajar con el *CSS* es algo engorroso y puede se complicado ya que se cargan en muchas plantillas del proyecto varios ficheros de *CSS* y en algunos casos pueden repetirse clases y que el resultado final no se muestre correctamente.
- El segundo problema radicaba en la búsqueda de usuarios ya si se realizaba una consulta a la base de datos con el string introducido en el buscador en general el resultado obtenido de la consulta era vacío. Por ello se introdujo los indices *fulltext* sobre aquellos campos en los que se pretendía consultar. Esta era la solución más optima debido a que los Bundle que realizaban dicha tarea eran difíciles de instalar por la compatibilidad de versiones con los distintos componentes software del proyecto, mientras que los índices solo eran una modificación sobre la base de datos y los resultados finales de las búsquedas eran mejores.

5.5.3. Burn down chart

Una vez finalizado el último sprint no quedaban historias de usuario que realizar por lo que el burn down chart resultante muestra el desarrollo de story points de todo el proyecto frente al desarrollo ideal. Se observa que la gráfica varía y es mucho más escalonada que el ideal debido a que los story points de una tarea se contabilizaban de una cuando se terminaba la tarea y no como en el ideal que cada día considera que has hecho una cantidad fija de story points.

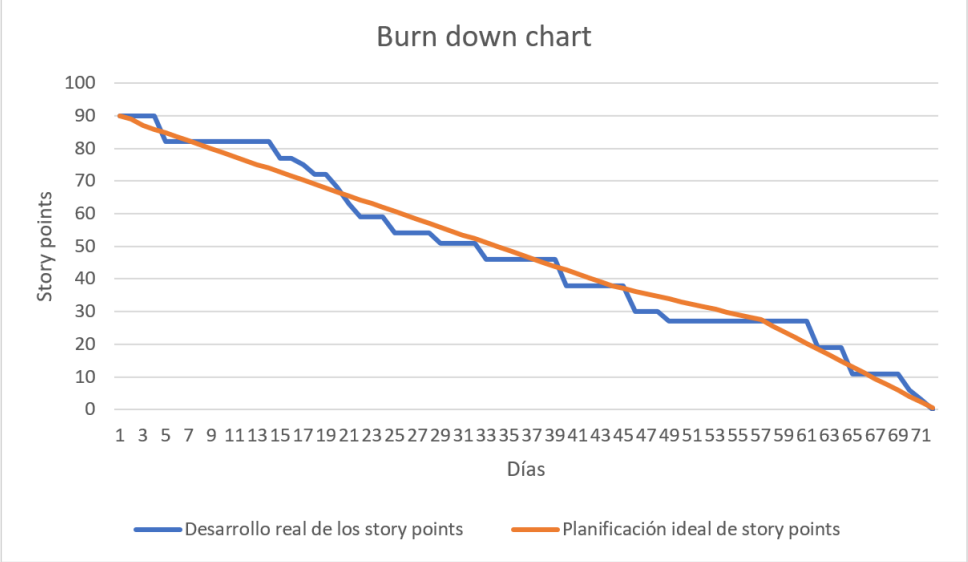

Figura 5.6: *Burn down chart final tras el quinto sprint*

Capítulo 6

Resultado final

Este capítulo muestra el resultado final de las distintas pantallas implementadas en Educeca durante el desarrollo de la estancia en prácticas. Primero se mostraran las imágenes de las pantallas de la biblioteca (Figuras 6.1, 6.2, 6.3, 6.4 y 6.5).

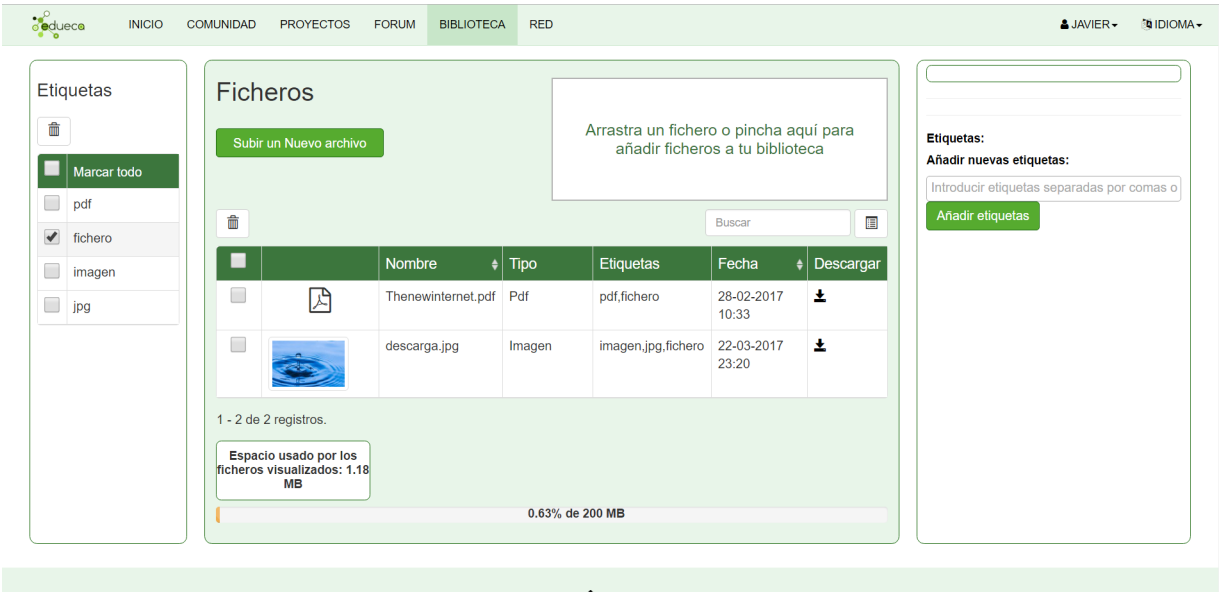

Figura 6.1: *Pantalla de biblioteca*

El resultado cuando alguien selecciona un fichero se muestra en la figura 6.2 donde se carga la información sobre el fichero en la parte derecha y permite cambiar el nombre, eliminar etiquetas o añadir nuevas.

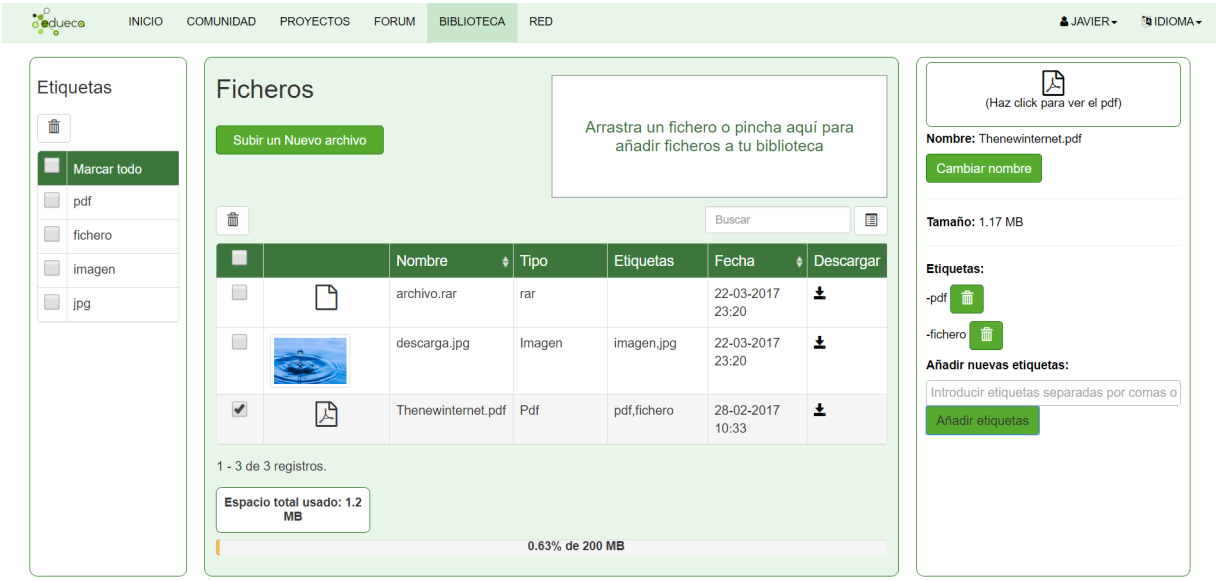

Figura 6.2: *Pantalla de biblioteca con un fichero seleccionado*

Tras seleccionar una imagen aparece en el panel lateral la imagen en un tamaño pequeño. Si se hace click sobre la imagen aparece una ventana modal donde aparece la imagen a mayor tamaño como muestra la figura 6.3.

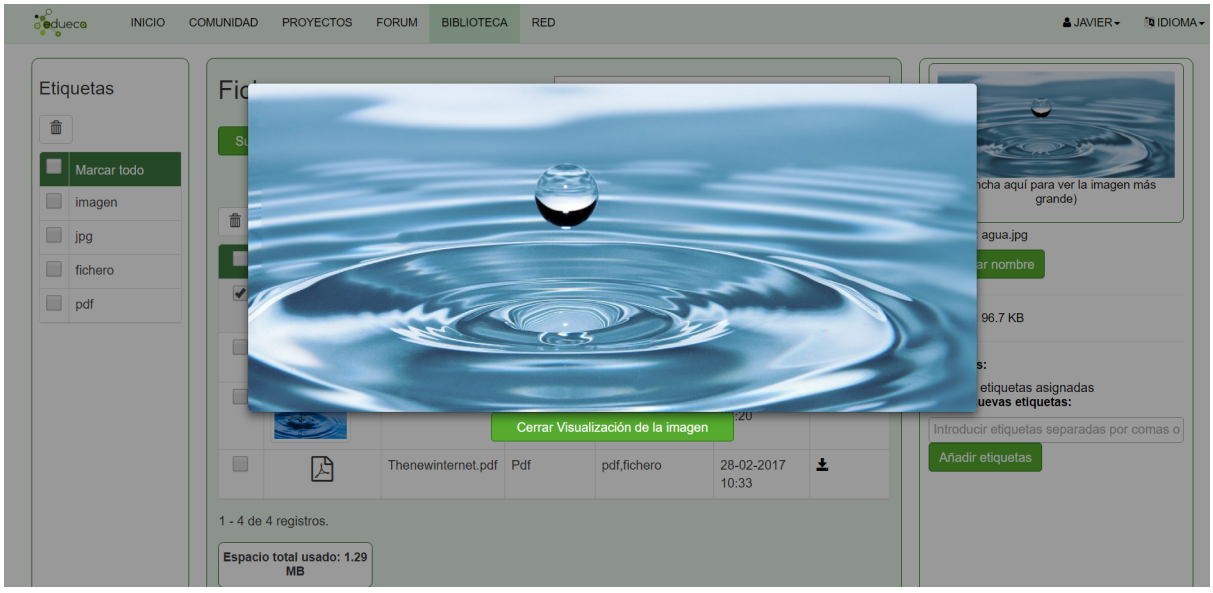

Figura 6.3: *Pantalla de biblioteca, visualización de imagen*

De una forma similar a las imágenes, al seleccionar un PDF el sistema permite hacer click sobre el icono que aparece en el panel lateral. Una vez se hace click sobre el icono aparece una ventana modal donde se visualiza el contenido del PDF como muestra la figura 6.4.

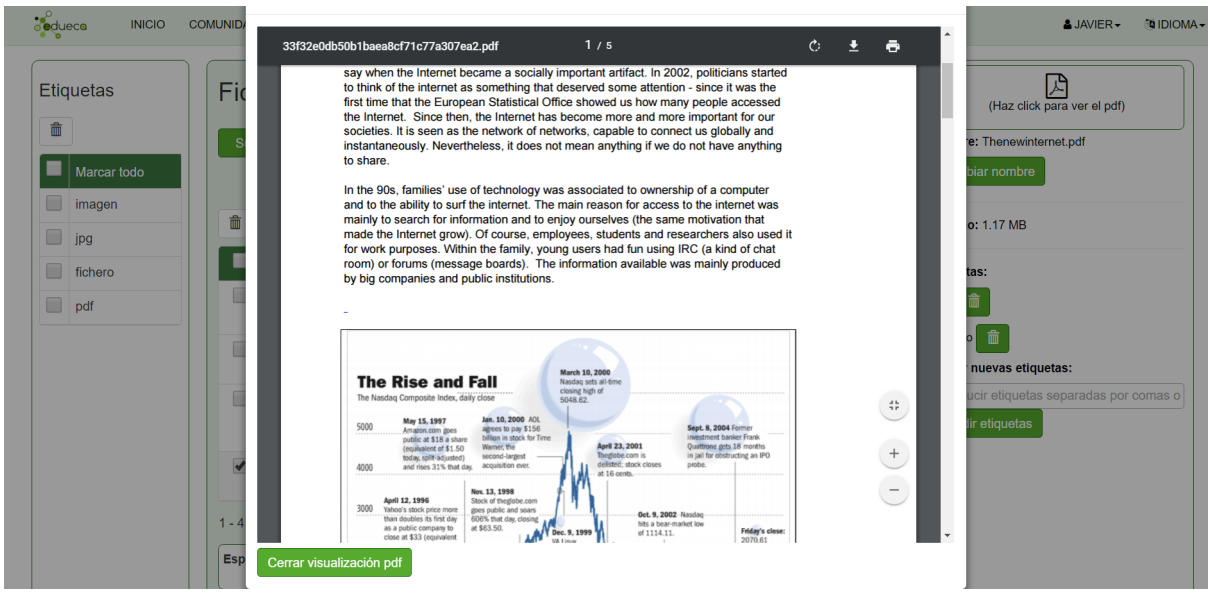

Figura 6.4: *Pantalla de biblioteca, visualización de pdf*

La última pantalla del apartado de Biblioteca (figura 6.5) consiste en la pantalla que permite subir un fichero, modificar el nombre del fichero a gusto del usuario y si el usuario lo considera necesario añadirle etiquetas al fichero que se pretende subir a Edueca.

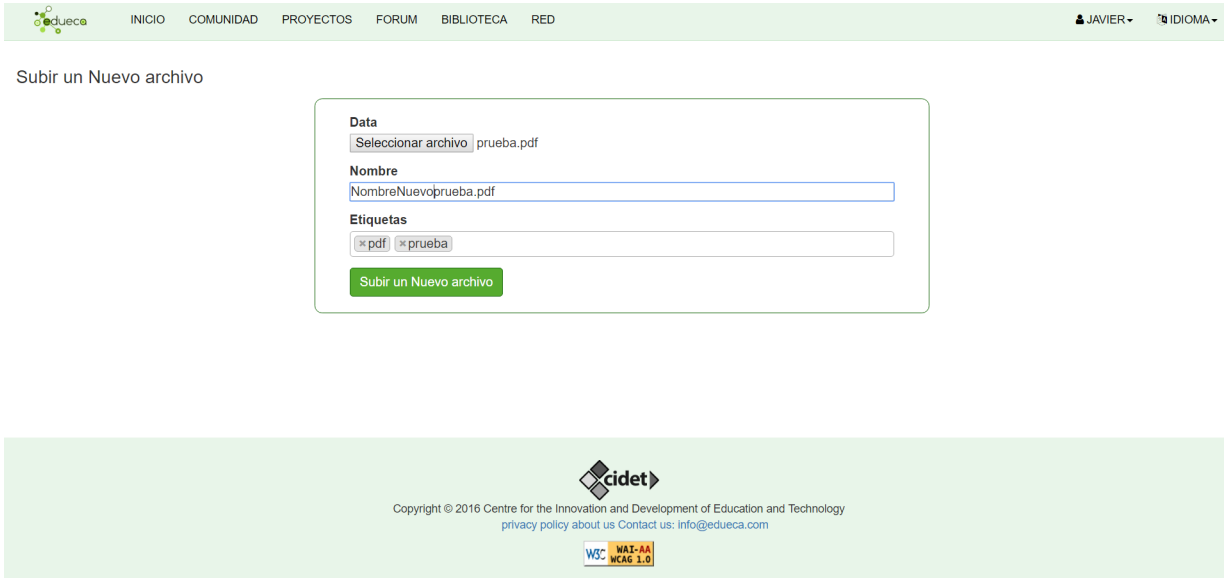

Figura 6.5: *Pantalla de biblioteca, subir nuevo fichero*

A continuación se muestran las figuras del apartado de Red. La figura 6.6 muestra la pantalla inicial donde se pueden leer los mensajes publicados por los usuarios añadidos como amigos y los mensajes publicados por el propio usuario.

Figura 6.6: *Pantalla de red, mensajes de amigos*

Por otro lado existe una pantalla donde el usuario puede ver su muro personal. Este muro personal es una pantalla donde solo se muestran los mensajes publicados por el propio usuario como muestra la figura 6.7.

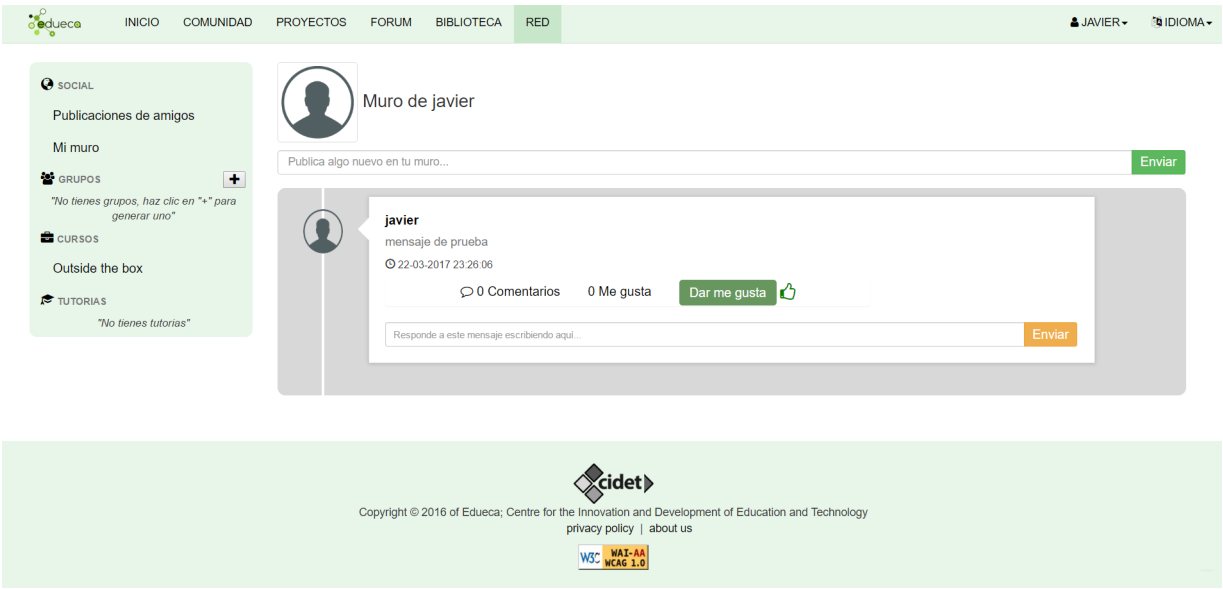

Figura 6.7: *Pantalla de red, muro personal del usuario*

Cuando un usuario utiliza el buscador que hay en la página principal de la Red se muestra una ventana modal con los resultados de la búsqueda como muestra la figura 6.8.

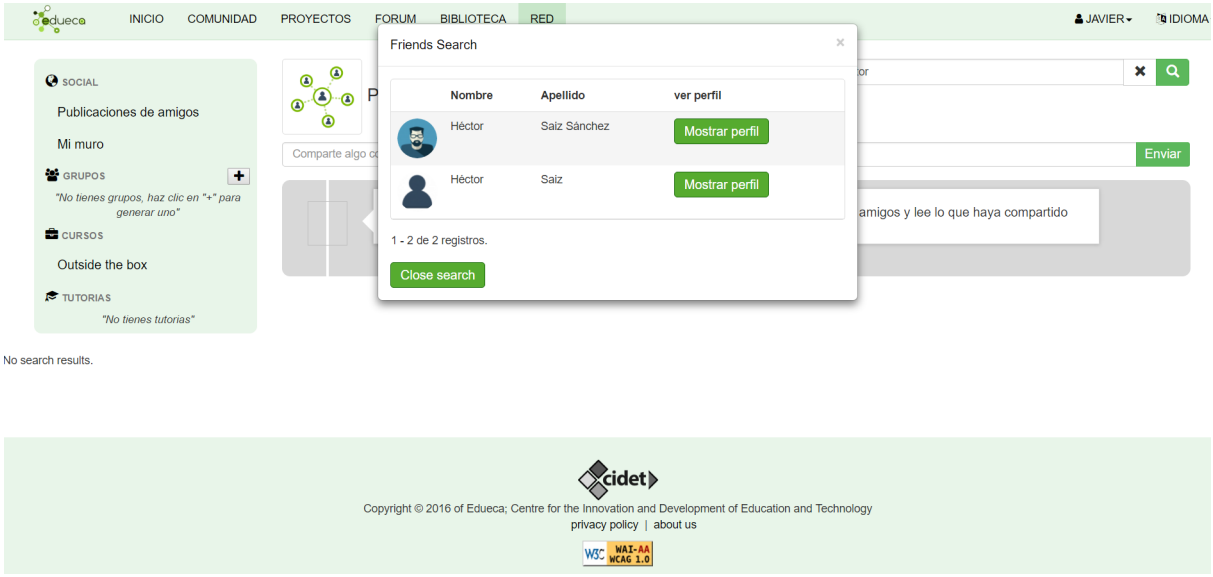

Figura 6.8: *Pantalla de red, buscar usuario*

Cuando el usuario utiliza el buscador y selecciona se muestra una pantalla con el muro personal de dicho usuario como aparece en la figura 6.9

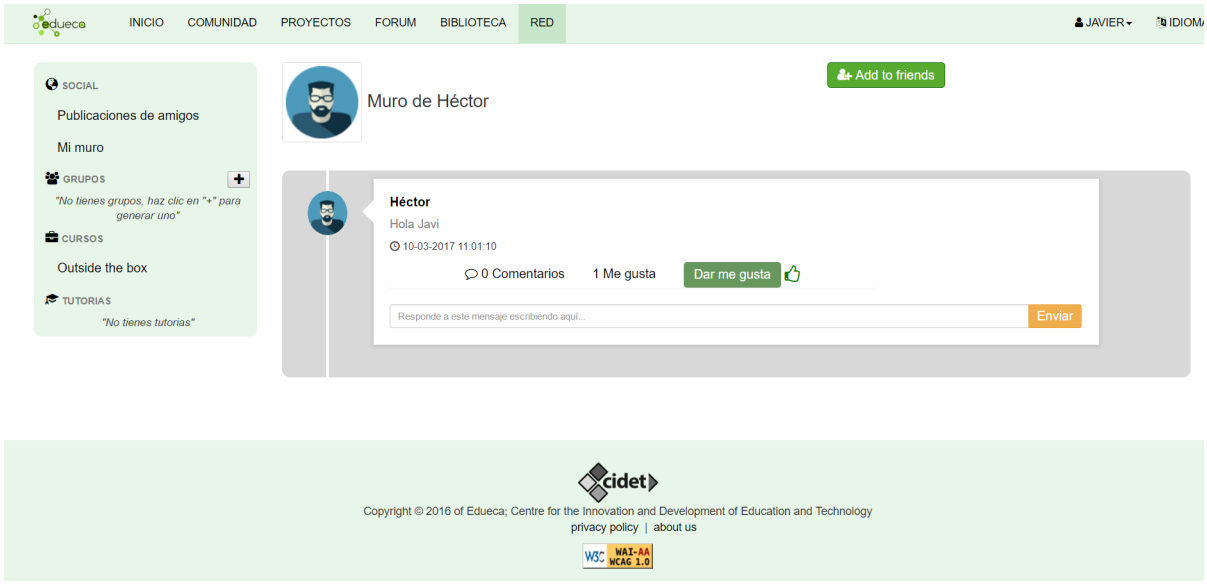

Figura 6.9: *Pantalla de red, muro de otro usuario con opción de añadir usuario como amigo*

En el caso de que el usuario seleccionado sea ya amigo del usuario entonces se muestra un botón en el muro del usuario seleccionado para poder eliminar dicho usuario como amigo (ver en figura 6.10).

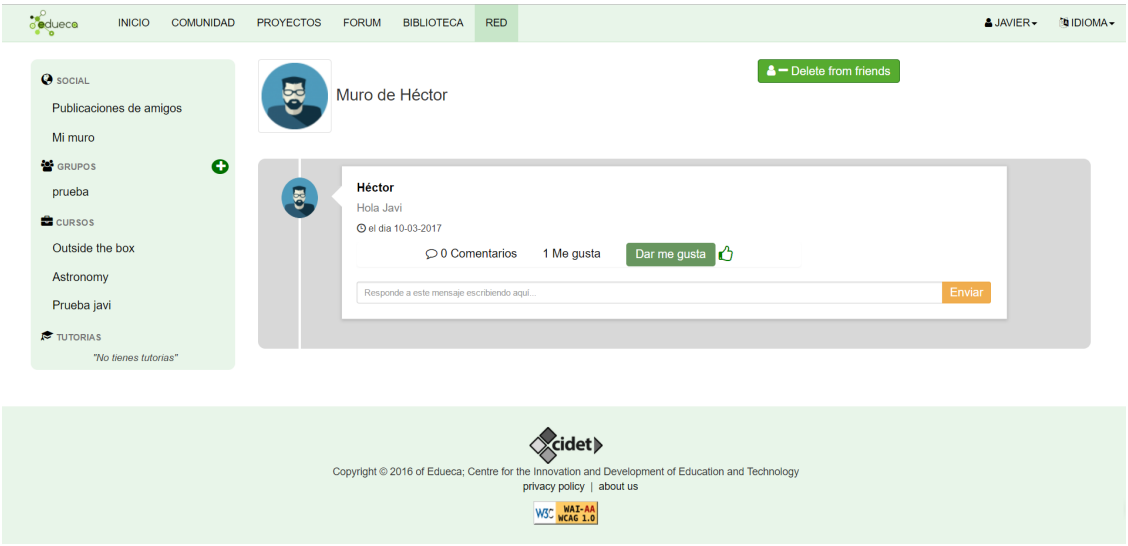

Figura 6.10: *Pantalla de red, muro de otro usuario con opción de eliminar usuario como amigo*

La figura 6.11 muestra un contenido de un proyecto en el que aparece un botón con un icono de un libro y una cruz. Este botón permite al usuario generar un PDF con el contenido que se está visualizando y añadirlo a la Biblioteca.

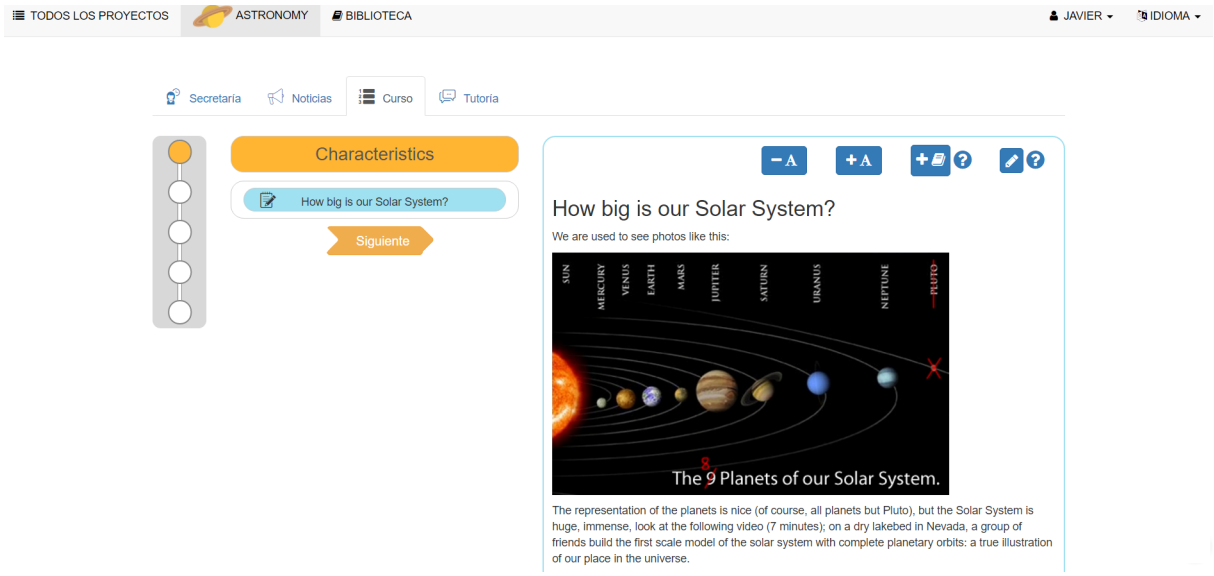

Figura 6.11: *Contenido de un curso con opción de añadir a la biblioteca*

Capítulo 7

Conclusiones

En este capítulo se muestran los distintos tipos de conclusiones obtenidas tras la finalización de la estancia en prácticas en la empresa CIDET.

7.1. Conclusiones formativas

Durante toda la estancia en prácticas he obtenido formación en diversos lenguajes de programación o tecnologías que o bien no había aprendido durante la carrera o bien solo tenía una ligera idea sobre su funcionamiento. No sabía nada por ejemplo de Symfony o Twig y al termino de las prácticas considero que podría realizar todo un proyecto sin mucha dificultad gracias a la formación obtenida a través de los manuales facilitados y la ayuda de mi supervisor de prácticas. Además la experiencia obtenida con la mezcla en un proyecto de diferentes tecnologías es algo que considero que no se puede obtener, o es muy complicado, en unas prácticas de clase.

Toda esta formación obtenida puede ser de gran utilidad en un futuro profesional ya que son tecnologías que tienen un gran uso.

7.2. Conclusiones personales

Estoy profundamente satisfecho con la estancia en prácticas tanto por los objetivos logrados, como por los conocimientos adquiridos como por el trato recibido por todo el personal de CIDET.

Ha sido un lugar muy cómodo donde trabajar, además considero que se han tenido en cuenta mis opiniones e ideas no solo en el desarrollo del proyecto en prácticas sino también en consultas que se me han realizado sobre otros aspectos de Edueca ajenos al proyecto de la estancia en prácticas. Creo que es de agradecer que se tengan en cuenta las opiniones o el trabajo de un estudiante ya que es algo que no se cumple en todas las empresas.

Bibliografía

[1] AJAX http://api.jquery.com/jquery.ajax/ (último acceso: 5/03/2017)

[2] BITBUCKET https://bitbucket.org/ (último acceso: 12/03/2017)

[3] BOOTSTRAP http://getbootstrap.com/ (último acceso: 12/03/2017)

[4] BOOTSTRAP W3 https://www.w3schools.com/bootstrap/ (último acceso: 12/03/2017)

[5] CSS https://developer.mozilla.org/es/docs/Web/CSS (último acceso: 1/03/2017)

[6] DOCTRINE http://www.doctrine-project.org (último acceso: 22/03/2017)

[7] ECMASCRIPT 5.1 https://www.ecma-international.org/ecma-262/5.1/ (último acceso: 8/03/2017)

[8] EXPLICACION BASE64 https://blogs.oracle.com/rammenon/entry/base64_explained (último acceso: 16/04/2017)

[9] HTML https://developer.mozilla.org/es/docs/Web/HTML (último acceso: 1/03/2017)

[10] JAVASCRIPT https://www.w3schools.com/js/js_intro.asp (último acceso: 26/02/2017)

[11] JIRA https://es.atlassian.com/software/jira (último acceso: 26/03/2017)

[12] MIME TYPE https://www.sitepoint.com/web-foundations/mime-types-complete-list/ (último acceso: 15/04/2017)

[13] MYSQL https://dev.mysql.com/doc/ (último acceso: 3/03/2017)

[14] PHP http://php.net/manual/es/intro-whatis.php (último acceso: 23/02/2017)

[15] PHPSTORM https://www.jetbrains.com/phpstorm/ (último acceso: 18/02/2017)

[16] METODOLOGÍA SCRUM http://comunidad.iebschool.com/iebs/general/metodologia-scrum/ (último acceso: 7/03/2017)

[17] METODOLOGÍA AGIL SCRUM https://proyectosagiles.org/que-es-scrum/ (último acceso: 3/03/2017)

[18] SMGL http://www.hipertexto.info/documentos/sgml.htm (último acceso: 15/04/2017)

[19] STORY POINTS https://johnnyordonez.com/2014/11/25/que-mismo-es-un-story-point/ (último acceso: 26/03/2017)

[20] Symfony http://symfony.com/legacy/doc/jobeet/1_3/es/04?orm=Propel (último acceso: 21/03/2017)

[21] Symfony 2.4, el libro oficial https://librosweb.es/libro/symfony_2_x/ (último acceso: 25/03/2017)

[22] Twig para diseñadores de plantillas http://gitnacho.github.io/Twig/templates.html (último acceso: 19/03/2017)

[23] Twig http://twig.sensiolabs.org/ (último acceso: 11/03/2017)

[24] World Wide Web (W3C) http://www.w3c.es/ (último acceso: 15/04/2017)

Anexo A

Introducción a Symfony

Lo primero en Symfony es generar un Bundle, un Bundle es un conjunto estructurado de archivos que se encuentran en un directorio y que implementan una sola característica. Incluye todos los tipos de ficheros relacionados con una función específica, incluyendo clases *PHP*, configuración, e incluso hojas de estilo y archivos de *Javascript*.

A.1. Estructura de un bundle

Los elementos comunes en un bundle son:

- *Controller/*: contiene los controladores del Bundle.
- *Resources/config/*: contiene la configuración, incluyendo la configuración de enrutamiento (por ejemplo, routing.yml).
- *Resources/views/*: contiene las plantillas organizadas según el nombre del controlador (por ejemplo, Hello/index.html.twig).
- *Resources/public/*: contiene recursos web (imágenes, hojas de estilo, etc.).

Una vez generado el Bundle para trabajar con Symfony hay que tener en cuenta que se basa en el modelo vista controlador, los controladores contienen acciones que se ejecutan cuando desde el navegador va a una determinada página.

Para definir una pagina en Symfony hay que seguir dos pasos:

- Definir una ruta: se define una URL o una estructura de URL con la cual se pretende que cuando el usuario vaya a esa página o estructura entonces se cargue la página deseada.
- crear un controlador y mapear una acción con la URL definida anteriormente, de forma que al acceder a la URL se ejecutará el código de la acción y se responde al usuario mediante un objeto Response.

A.2. Controlador

Un controlador se trata de una función PHP que obtiene la información de una petición HTTP(Request) y genera y devuelve la respuesta HTTP(Response). Contiene toda la lógica que la aplicación necesita para generar el contenido de la página, y la respuesta puede ser una página HTML, un documento XML, un array JSON serializado, una cabecera de error, etc.

Un ejemplo de controlador sería el siguiente:

```
<?php

namespace AppBundle\Controller;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;

class ExampleController extends Controller
{
    /**
     * @Route("/", name="/ejemplo")
     */
    public function exampleAction(Request $request)
    {
        return $this->render('example.html.twig');
    }
}
```

Este controlador de ejemplo ejecuta la función exampleAction que renderiza la plantilla Twig 'example.html.twig'.

Sin embargo desde un controlador pueden realizarse muchas más acciones además de renderizar una plantilla Twig:

- Devolver código HTML

```
return new Response('<html><body>Ejemplo</body></html>');
```

- Recoger parámetros de la ruta como argumentos

```
/**
 * @Route("/ejemplo/{nombre}", name="ejemplo")
 */
public function ejmploAction($nombre)
{
    return new Response('<html><body>Ejemplo de '.$nombre.'!</body></html>');
}
```

- Crear respuesta de tipo JSON

```
/**
 * @Route("/ejemplo/{nombre}", name="ejemplo")
 */
public function ejmploAction($nombre)
{
  $response = new Response(json_encode(array('nombre' => $nombre)));
  $response->headers->set('Content-Type', 'application/json');
```

- Redirigir a una URL o otro método del controlador

```
return $this->redirectToRoute('ejemplo2'); // redirigir a otro metodo

return $this->redirect('http://ejemplo.com'); // redirigir a una url
```

- Renderizar una plantilla y mandar variables a la misma

```
return $this->render('app/ejemplo.html.twig', array('var1'=>$var1,
'var2' => var2));
```

A.3. Plantilla twig

Twig es un motor de plantillas desarrollado para el lenguaje de programación *PHP* y que nace con el objetivo de facilitar a los desarrolladores de aplicaciones web que utilizan la arquitectura MVC.

Ejemplo de una plantilla twig:

```
<html>
<head>
<title>Ejemplo de Twig</title>
</head>
<body>
<h1>{{ titulo }}</h1>
<ul id="navigation">
{% for elem in listaValores %}
<li><a href="{{ elem.url }}">{{ elem.texto }}</a></li>
{% endfor %}
</ul>
</body>
</html>
```

Se observa que a través de los valores que se envían a la plantilla se genera una página con un contenido u otro.

En este ejemplo se observan dos sintaxis que aparecen:

- {{...}}: Se utiliza para imprimir el valor de una variable por pantalla.
- {%...%}: Se trata de una etiqueta que controla la lógica de la plantilla. Es utilizado cuando tengamos que utilizar bucles, estructuras if-else, for, etc.

A.4. Doctrine

Doctrine es un mapeador de objetos-relacional (ORM) escrito en PHP que proporciona una capa de persistencia para objetos PHP. Es una capa de abstracción que se sitúa justo encima de un SGBD.

Esto significa que Symfony trabaja con objetos formados por la información de la base de datos y después Doctrine gestiona estos objetos para actualizar la base de datos.

Para trabajar con estos objetos se generan entidades, estas entidades son clases de PHP que doctrine relaciona con una tabla de la base de datos.

Por ejemplo si en la base de datos existe una tabla usuario entonces en Symfony debe existir una clase usuario de la siguiente forma:

```
namespace Acme\StoreBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity
 * @ORM\Table(name="usuario")
 */
class Usuario
{
    /**
     * @ORM\Id
     * @ORM\Column(type="integer")
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    protected $id;

    /**
     * @ORM\Column(type="string", length=100)
     */
    protected $nombre;
}
```

Esta clase deberá contener los getters y setters de cada uno de sus atributos.

A partir de una entidad se puede generar la tabla en la base de datos ejecutando desde la consola:

```
$     app/console doctrine:schema:update --force
```

A.4.1. Acciones sobre los objetos de una entidad

Nuevo objeto

Para generar un nuevo objeto de entidad usuario definida anteriormente seria necesario ejecutar:

```
$usuario = new Usuario();
$usuario->setNombre('A␣Foo␣Bar');

$em = $this->getDoctrine()->getManager();
$em->persist($usuario);
$em->flush();
```

Obtener objeto

Para obtener un objeto usuario a través de una variable $id:

```
 $usuario = $this->getDoctrine()
->getRepository('EjemploBundle:Usuario')
->find($id);
```

Para obtener todos los objetos de la entidad:

```
 $usuario = $this->getDoctrine()
->getRepository('EjemploBundle:Usuario')
->findAll();
```

Actualizar un objeto

```
$em = $this->getDoctrine()->getManager();
$usuario = $em->getRepository('EjemploBundle:Usuario')->find($id);

$usuario->setNombre('Nuevo␣nombre');
$em->flush();
```

Eliminar un objeto

```
$em = $this->getDoctrine()->getManager();
$usuario = $em->getRepository('EjemploBundle:Usuario')->find($id);

$em->remove($usuario);
$em->flush();
```

Anexo B

Herramientas utilizadas

B.1. Select2

Se trata de un fichero *jQuery* que busca reemplazar las cajas de selección habituales, ofrece un cuadro de selección personalizable con soporte para buscar, etiquetar, añadir conjuntos de datos remoto y muchas otras opciones muy utilizadas.

Figura B.1: *Ejemplo de Select2*

B.2. Dropzone

DropzoneJS es una librería de código abierto en *JavaScript* que proporciona subidas de archivos drag'n'drop con vistas previas de imágenes.

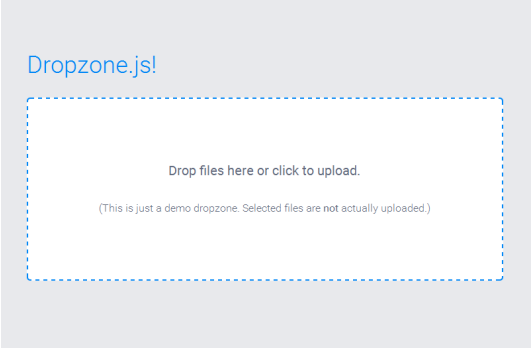

Figura B.2: *Ejemplo de dropzone*

B.3. Composer

Composer es un gestor de paquetes a nivel de aplicación para el lenguaje de programación PHP que proporciona un formato estándar para gestionar las dependencias del software PHP y las librerías necesarias. Realiza una instalación local para cualquier proyecto, las librerías se instalan en un directorio por defecto (normalmente es /vendor). *Composer* es capaz de instalar las librerías que requiere el proyecto con las versiones que necesiten.

En el caso de este proyecto se utiliza para la instalación de Bundles de terceros en Symfony.

Figura B.3: *Logo de Composer*

B.4. ckeditor

Se trata de un editor de texto que introduce funcionalidades de procesador de texto en páginas web sin instalar ningún componente en la máquina del cliente.

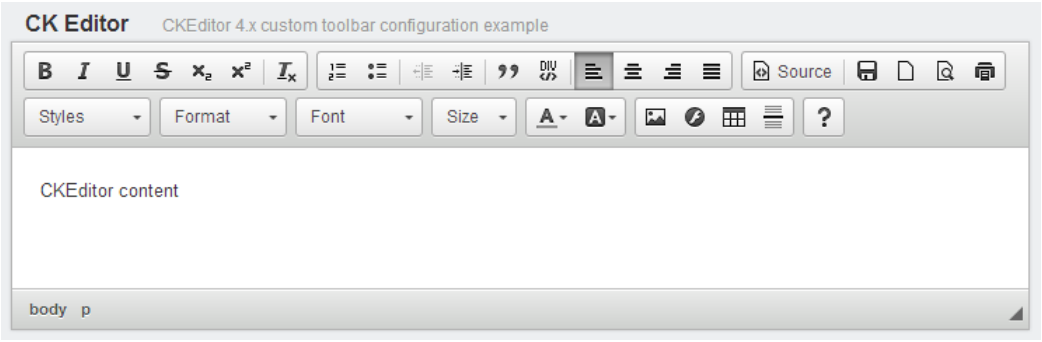

Figura B.4: *Ckeditor*