



UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto de Fin de Grado en Ingeniería Informática

**YourMOOC4all – Diseño inclusivo y  
retroalimentación útil de cursos MOOC para  
estudiantes con discapacidad**

Pedro Manuel Muñoz Morales

Dirigido por: Francisco Iniesto Carrasco

Codirigido por: Covadonga Rodrigo San Juan

Curso: 2017-2018: 1<sup>a</sup> Convocatoria





## **YourMOOC4all – Diseño inclusivo y retroalimentación útil de cursos MOOC para estudiantes con discapacidad**

Proyecto de Fin de Grado Modalidad Externo

Realizado por: Pedro Manuel Muñoz Morales

Dirigido por: Francisco Iniesto Carrasco

Codirigido por: Covadonga Rodrigo San Juan

Tribunal                    calificador

Presidente: D/Dª.

Secretario: D/Dª.

Vocal: D/Dª.

Fecha de lectura y defensa:

Calificación:



# Agradecimientos

En primer lugar, quiero dar las gracias a mi mujer Marga y a mis hijos, Andrea y Pablo, sin la cobertura permanente me han dado no hubiera sido posible lograrlo.

También quiero agradecer el apoyo de mis amigos y compañeros del trabajo, que han hecho el papel de compañeros de universidad, soportando mis quejas y lamentos.

Sin olvidar a Doña Covadonga Rodrigo San Juan y a Don Francisco Iniesto Carrasco, los cuales me han prestado toda su atención y apoyo durante la elaboración de este proyecto.

Y en general a todos los docentes de la UNED que han hecho posible que yo haya llegado hasta el final de la carrera.



# Resumen

El objetivo de este proyecto es la creación de un portal Web que sirva como sistema recomendador de cursos MOOC (cursos masivos y abiertos en línea) para usuarios con necesidades especiales, de forma que, dentro de la oferta en lengua española, el usuario pueda encontrar los cursos más idóneos y de mayor calidad que se adecúen a sus necesidades de accesibilidad. El portal recoge la oferta más reciente de cursos a partir de las plataformas más conocidas como UNED Abierta, edX, Coursera y MiriadaX y permite hacer búsquedas de forma categorizada, pormenorizando los resultados en función de la adecuación del curso a las necesidades de accesibilidad de los usuarios.

Para ello, el sistema recomendador muestra al usuario una lista de cursos MOOC al estilo de los portales Course Talk o MOOC - list, pero de forma más completa, incluyendo aspectos relacionados con la experiencia de uso, como son el nivel de accesibilidad y usabilidad, la relación con las tecnologías de asistencia, la disponibilidad de subtitulado en diversos idiomas, de lenguaje de signos, transcripciones y otros factores que influyen en el proceso de aprendizaje de estudiantes con discapacidad.

El portal YourMOOC4all dispone de una interfaz de usuario que se ha diseñado siguiendo las pautas y estándares que aseguran el mayor nivel de usabilidad y la accesibilidad. Así, se ha asegurado que puede ser utilizada por la mayor cantidad de personas, destacando por su claridad, facilidad de uso y navegación intuitiva.

# Abstract

The objective of this project is the creation of a Web portal that serves as a recommendation system for MOOC courses (mass and open online courses) for users with special needs, so that, within the offer in Spanish language, the user can find the most suitable and highest quality courses that fit your accessibility needs. The portal includes the most recent courses available from the most popular platforms such as UNED COMA, edX, Coursera and MiriadaX and allows searches in a categorized way, detailing the results according to the adaptation of the course to the accessibility needs of the students.

For this, the recommender system shows the user a list of MOOC courses in the style of the Course Talk or MOOC - list portals, but more completely, including aspects related to the user experience, such as the level of accessibility and usability, the relationship with assistance technologies, the availability of subtitling in various languages, sign language, transcriptions and other factors that influence the learning process of students with disabilities.

The YourMOOC4all portal has a user interface that has been designed following the guidelines and standards that ensure the highest level of usability and accessibility. Thus, it has been ensured that it can be used by the greatest number of people, standing out for its clarity, ease of use and intuitive navigation.

# Índice general

## Tabla de contenido

Agradecimientos .....	5
Resumen.....	7
Abstract .....	8
Índice general .....	9
Índice de figuras.....	13
Índice de tablas.....	15
Palabras clave .....	16
1. INTRODUCCIÓN.....	17
1.1 Motivación del proyecto .....	18
1.2 Objetivos del proyecto.....	19
1.3 Tecnologías utilizadas .....	20
1.3.1 Ruby, Ruby On Rails y sus librerías .....	21
1.3.2 HTML, CSS y JavaScript.....	22
1.3.3 Servidores de bases de datos .....	25
1.3.4 jQuery, Bootstrap y Chartkick .....	26
1.4 Estructura de la memoria .....	28
2. Análisis.....	29
2.1 Análisis de requisitos .....	29
2.1.1 Requisitos de tipos de usuarios o roles.....	29
2.1.2 Requisitos de registro .....	30
2.1.3 Requisitos de interfaz de búsquedas .....	30
2.1.4 Requisitos de Harvesting.....	30
2.1.5 Requisitos de interfaz de resultados.....	31
2.2 Modelo del dominio .....	31
2.3 Diagrama de casos de uso.....	33
2.4 Listado de casos de uso .....	35
2.4.1 Caso de uso: Buscar curso en el sistema.....	35
2.4.2 Caso de uso: Cambiar idioma del sistema.....	36
2.4.3 Caso de uso: Registrarse en el sistema .....	37
2.4.4 Caso de uso: Identificarse en el sistema .....	38

2.4.5 Caso de uso: Recuperar contraseña .....	39
2.4.6 Caso de uso: Cambiar contraseña .....	41
2.4.7 Caso de uso: Votar un curso.....	42
2.4.8 Caso de uso: Seleccionar un curso como interesante .....	44
2.4.9 Caso de uso: Adquirir información en las diferentes plataformas .....	45
2.4.10 Caso de uso: Gestionar los cursos .....	47
2.4.11 Caso de uso: Gestionar las Instituciones, Plataformas e Idiomas .....	49
2.4.12 Caso de uso: Gestionar edición anterior .....	50
2.4.13 Caso de uso: Gestionar los usuarios .....	51
2.4.14 Caso de uso: Adquirir y analizar cursos MOOC .....	53
2.5 Modelo conceptual.....	56
2.6 Requisitos funcionales.....	58
2.7 Requisitos de seguridad.....	58
2.8 Requisitos no funcionales .....	59
<b>3. Diseño .....</b>	<b>61</b>
<b>3.1 Arquitectura del sistema.....</b>	<b>61</b>
<b>3.1.1 Modelo Cliente-Servidor .....</b>	<b>61</b>
<b>3.1.2 Modelo Vista Controlador (MVC) .....</b>	<b>62</b>
<b>3.1.3 Servicios REST .....</b>	<b>64</b>
<b>3.2 Guía de estilo y diseño.....</b>	<b>64</b>
<b>3.2.1 El esquema de página. ....</b>	<b>65</b>
<b>3.2.2 Accesibilidad.....</b>	<b>66</b>
<b>3.2.3 Diseño del interfaz y contenidos de la página.....</b>	<b>66</b>
<b>3.3 Diseño de interfaces de usuario.....</b>	<b>66</b>
<b>3.3.1 Home.....</b>	<b>66</b>
<b>3.3.2 Administrar.....</b>	<b>67</b>
<b>3.3.3 Cursos.....</b>	<b>67</b>
<b>3.3.4 Evaluación .....</b>	<b>68</b>
<b>3.3.5 Inicio.....</b>	<b>68</b>
<b>3.3.6 Información, detalle de un curso.....</b>	<b>69</b>
<b>3.4 Diagrama de navegación.....</b>	<b>69</b>
<b>3.5 Diagrama de clases .....</b>	<b>70</b>
<b>3.6 Diseño de la persistencia .....</b>	<b>72</b>
<b>3.6.1 Diagrama de Entidad Relación .....</b>	<b>73</b>

4.	Implementación .....	77
4.1	Partes fundamentales.....	77
4.1.1	Adquisición de información o Harvesting .....	77
4.1.2	Búsqueda de cursos MOOC.....	81
4.1.3	Evaluación de los cursos .....	83
4.2	Esquema de la base de datos.....	84
4.3	Ejemplo de funcionamiento del paradigma MVC .....	85
4.4	Árbol de directorios de la aplicación .....	87
4.5	Control de versiones.....	89
4.5.1	Git .....	89
4.5.2	GitHub .....	92
4.6	Alojamiento en la web.....	92
4.7	Tarea diaria .....	95
5.	Pruebas .....	97
5.1	Pruebas unitarias.....	98
5.2	Pruebas funcionales .....	99
5.3	Pruebas de integración .....	100
5.4	Pruebas de usabilidad.....	100
5.5	Pruebas de accesibilidad.....	102
5.6	Pruebas de seguridad .....	103
5.7	Pruebas manuales .....	105
5.8	Otras pruebas.....	106
6	Planificación.....	107
6.1	Modelo incremental .....	107
6.2	Planificación temporal.....	108
7	Conclusiones y trabajos futuros .....	111
7.1	Conclusiones .....	111
7.2	Trabajos futuros .....	112
8	Manual.....	115
8.1	Página de inicio .....	115
8.2	Resultado búsqueda .....	116
8.3	Página de registro.....	117
8.4	Página de identificación.....	117
8.5	Recuperación de contraseña.....	118

8.6 Ver información sobre un curso .....	119
8.7 Inicio para usuario registrados e identificados .....	120
8.8 Editar información de usuario.....	121
8.9 Evaluar curso .....	121
8.10 Página de inicio para el administrador .....	122
8.11 Gestión de cursos .....	123
8.12 Editar curso .....	123
8.13 Añadir/modificar/eliminar idiomas .....	125
REFERENCIAS BIBLIOGRÁFICAS.....	127

# Índice de figuras

Figura 1 - Modelo de dominio.....	32
Figura 2 -Diagrama de casos de uso.....	34
Figura 3 - caso de uso 1 .....	36
Figura 4 - caso de uso 2 .....	37
Figura 5 - caso de uso 3 .....	38
Figura 6 - caso de uso 4 .....	39
Figura 7 - caso de uso 5 .....	41
Figura 8 - caso de uso 6 .....	42
Figura 9 - caso de uso 7 .....	44
Figura 10 - caso de uso 8 .....	45
Figura 11 - caso de uso 9 .....	47
Figura 12 - caso de uso 10 .....	48
Figura 13 - caso de uso 11 .....	50
Figura 14 - caso de uso 12 .....	51
Figura 15 - caso de uso 13 .....	53
Figura 16 - caso de uso 14 .....	55
Figura 17 - Modelo conceptual 1 .....	56
Figura 18 - Modelo conceptual 2 .....	57
Figura 19 - Cliente-servidor.....	62
Figura 20 - modelo/vista/controlador .....	63
Figura 21 -Diseño - Home .....	67
Figura 22 – Diseño - administrar .....	67
Figura 23 – Diseño - cursos .....	68
Figura 24 – Diseño - evaluación .....	68
Figura 25 – Diseño – Inicio del usuario registrado .....	69
Figura 26 – Diseño – información del curso .....	69
Figura 27 - Diagrama de navegación .....	70
Figura 28 - Diagrama de clases .....	71
Figura 29 - Diagrama de minería.....	72
Figura 30 - Diagrama entida relación .....	74
Figura 31 - Diagrama del modelo.....	75
Figura 32 - Diagrama ActiveRecord .....	76
Figura 33 - Clase Minería .....	78
Figura 34 - Clase Coursera .....	79
Figura 35 - Método buscar_cursos.....	81
Figura 36 - Método puntuación .....	83
Figura 37 – esquema- tabla cursos.....	84
Figura 38 - esquema – tabla evoluciones .....	84

Figura 39 - ejemplo mvc 1.....	85
Figura 40 - ejemplo mvc 2.....	86
Figura 41 - ejemplo mvc 3.....	87
Figura 42 - árbol de directorios.....	87
Figura 43 - commits de Git.....	91
Figura 44 – GitHub.....	92
Figura 45 - Captura de pantalla con la cuenta de alojamiento en Heroku.....	93
Figura 46 – Configuración de Heroku.....	93
Figura 47 – Log de Heroku.....	94
Figura 48 - tarea .....	95
Figura 49 - código tarea .....	96
Figura 50 - pruebas funcionales .....	99
Figura 51 – eXaminator .....	102
Figura 52 - Achecker .....	103
Figura 53 - OWASP TOP 10 .....	104
Figura 54 - OWASP Zed Attack Proxy 1.....	104
Figura 55 - OWASP Zed Attack Proxy 2.....	105
Figura 56 - exploración de grafo en anchura .....	105
Figura 57 - modelo incremental.....	107
Figura 58 - LOCs.....	109
Figura 59 - Progressive Web Apps 1.....	112
Figura 60 - Progressive Web Apps 2.....	113
Figura 61 - Progressive Web Apps 3.....	113
Figura 62 - manual - inicio.....	115
Figura 63 - manual – búsqueda 1 .....	116
Figura 64 - manual - registro.....	117
Figura 65 - manual - identificación.....	117
Figura 66 - manual – recuperar contraseña 1.....	118
Figura 67 - manual – recuperar contraseña 2.....	118
Figura 68 - manual – recuperar contraseña 3.....	119
Figura 69 – manual – información del curso 1.....	119
Figura 70 - manual - información del curso 2 .....	120
Figura 71 – manual – inicio de usuario identificado .....	120
Figura 72 - manual – información de usuario .....	121
Figura 73 - manual – evaluar curso .....	121
Figura 74 - manual – inicio administrador .....	122
Figura 75 - manual - gestión de cursos.....	123
Figura 76 - manual – editar curso 2.....	123
Figura 77 - manual – editar curso 2.....	124
Figura 78 - manual – lista enlazada .....	124
Figura 79 - manual - idiomas.....	125

# Índice de tablas

Tabla 1 - Bootstrap .....	27
Tabla 2 - caso de uso 1.....	35
Tabla 3 - caso de uso 2.....	36
Tabla 4 - caso de uso 3.....	37
Tabla 5 - caso de uso 4.....	39
Tabla 6 - caso de uso 5.....	40
Tabla 7 - caso de uso 6.....	42
Tabla 8 - caso de uso 7.....	43
Tabla 9 - caso de uso 8.....	45
Tabla 10 - caso de uso 9.....	46
Tabla 11 - caso de uso 10.....	48
Tabla 12 - caso de uso 11.....	49
Tabla 13 - caso de uso 12.....	51
Tabla 14 - caso de uso 13.....	52
Tabla 15 - caso de uso 14.....	54
Tabla 16 - tareas rake .....	98
Tabla 17 - planificación temporal.....	108

# **Palabras clave**

MOOC – Massive On-line Open Courses

COMA - Cursos Online Masivos y Abiertos

Usabilidad y Accesibilidad

Harvesting

Diseño Universal de Aprendizaje

Framework Ruby On Rails

Gemas

Control de versiones

Git

GitHub

Heroku

# 1. INTRODUCCIÓN

Los MOOC (Massive Open Online Courses) o Cursos Online Masivos y Abiertos [1] [2], son una nueva forma de adquirir conocimientos ofrecidos a través de Internet de forma visual, muy interactiva y con una elevada componente de aprendizaje social. Los MOOC se apoyan en las mismas herramientas tecnológicas utilizadas en los entornos e-learning, pero al estar en abierto tienen mucha mayor difusión. También cuentan con un gran número de cursos en los que se desarrollan actividades con propuestas orientadas a la participación y la colaboración de los alumnos.

Estos cursos tienen la particularidad de que son muy intuitivos, en los que el usuario debe ser capaz de seguir la dinámica del curso de una manera aislada. La inscripción al mismo se hace de forma autónoma y se solicita al participante pocos datos, habitualmente su nombre, cuenta de correo, lugar de origen y nivel de estudios. El fenómeno MOOC está íntimamente ligado a otros dos fenómenos: el auge de los contenidos publicados en abierto (REAs – Recursos Educativos Abiertos) y el aprendizaje social a través del conectivismo y el constructivismo de conocimiento.

Los MOOC actualmente cuenta con millones de participantes en todo el mundo en plataformas como Coursera, UNED Abierta, edX, MiriadaX, etc. Los MOOC ofrecen cursos en distintas disciplinas, algunos sobre temas muy concretos y de índole técnico y otros, sin embargo, enfocados en compartir aspectos de cultura general o entretenimiento, como ejemplo el curso para “Escuchar música del mundo” ofertado por la Universidad de Pennsylvania.

Este tipo de formación tiene sus ventajas e inconvenientes. La ventaja más llamativa es que el ritmo de aprendizaje lo marca el alumno y que las clases se pueden volver a visualizar tantas veces como se necesite, o incluso parar la reproducción del mismo para hacer una consulta sobre un tema que se ha introducido en dicha clase. Por el contrario, la desventaja más importante de los cursos MOOC es la falta de comunicación directa con el profesorado.

Estos cursos cuentan con gran cantidad materiales audiovisuales, como por ejemplo los Mini-Videos Docentes Modulares, con una duración aproximada de 10 minutos y que pueden pausarse por indicación del docente para realizar alguna tarea, además, suelen contener evaluaciones automáticas o entre pares de alumnos y foros para la comunicación entre alumnos y entre alumno y profesor.

Para que una enseñanza a distancia pueda ser considerada MOOC debe cumplir ciertos requisitos:

- Ser un curso, que contenga una estructura orientada al aprendizaje con evaluaciones para poder acreditar el conocimiento adquirido.
- Ser online, impartido a distancia y usando Internet como canal de comunicación. Por lo tanto, no debe haber coincidencia física, ni temporal, entre alumno y profesor.
- Ser masivo, los alumnos matriculados en un curso pueden ser ilimitados, ya que no deben estar concentrados en un aula.
- Ser global, esta característica implica que se puede seguir el curso desde cualquier ubicación geográfica.
- Estar abierto, el acceso a los materiales debe ser gratuita y accesible desde Internet. Esto último no siempre se cumple ya que hay plataformas que son de pago.

Para clasificar los cursos MOOC se puede realizar de varias formas, pero la más usual distingue dos tipos:

- Los MOOC conectivistas (cMOOC) que ponen el énfasis en la creación de conocimiento por parte de los estudiantes, en la creatividad y la autonomía, el aprendizaje social y colaborativo.
- Los MOOC extendidos (xMOOC) copian el modelo tradicional de enseñanza online, ponen el énfasis en la visualización de vídeos y la realización de pequeños ejercicios. Las plataformas comerciales utilizan esta metodología.

Todavía hoy la mayor problemática que sufren los cursos MOOC es la alta tasa de abandono, esta problemática está sustentada en dos pilares: el primero es que al ser gratuitos los alumnos se matriculan en él aun sabiendo que no lo realizarán, el segundo es que el alumno hace una estimación previa del tiempo necesario para dedicar a ese curso, después se da cuenta de que el cálculo no ha sido correcto y al no disponer de más tiempo para él lo abandona.

Actualmente la expansión de los cursos MOOC es gradual e imparable pero también, afortunadamente existe un aumento fehaciente de la calidad de los mismos. De momento los MOOCs han venido para quedarse y ya suponen un hito importante en la evolución de la educación.

## 1.1 Motivación del proyecto

Los MOOC son un éxito reciente de las metodologías de aprendizaje on-line, y en algunos casos se posicionan como una alternativa a los cursos tradicionales en la educación superior [2]. Esta nueva tipología de cursos ofrece a docentes, investigadores y profesionales la oportunidad de experimentar en el campo pedagógico y estudiar diferentes posibilidades para integrar sus elementos en el campus físico o virtual como formas de aula invertida (flipped-classroom) o aprendizaje combinado semi-presencial [3].

Los resultados del aprendizaje basado en estas metodologías despiertan mucho interés en el ámbito investigador y últimamente se han publicado numerosos estudios midiendo el impacto sobre las tasas de participación, retención y finalización en el aprendizaje de adultos que tiene el diseño pedagógico y visual de los MOOC, su arquitectura de información, el nivel de usabilidad y su diseño de interacción [4] [5].

La reciente incorporación a este nuevo aprendizaje abierto y en línea, la creación de nuevas formas educativas (tanto desde el punto de vista pedagógico como tecnológico) pueden utilizarse para repensar la educación y también para renovar la educación inclusiva que puede llegar a todos los ciudadanos. Alcanzar la inclusión social solo se puede lograr incorporando estrategias inclusivas, por ello la importancia de focalizar e incluir a los grupos vulnerables, como son las personas con discapacidad [6].

Los MOOCs ofrecen importantes beneficios a las personas con discapacidad, por su facilidad de acceso (basta un correo electrónico y un acceso a internet), no son necesarios desplazamientos fuera del domicilio y no tienen coste (o muy moderado para el caso de solicitar un certificado o badge). Sin embargo, la oferta actual de MOOCs adolece precisamente de la información básica que estos estudiantes necesitan para matricularse: su nivel de accesibilidad o su adecuación a las diversas tecnologías de apoyo que pueden utilizar los estudiantes con discapacidad (como lectores de pantalla, subtítulados, etc).

El Diseño Universal para el Aprendizaje (DUA) es un marco que aborda el principal obstáculo para promover aprendices expertos en los entornos de enseñanza: los currículos inflexibles, “talla-única-para-todos”. Son precisamente estos currículos inflexibles los que generan barreras no intencionadas para acceder al aprendizaje [7].

Hay tres principios fundamentales basados en la investigación neurocientífica que guían el DUA y proporcionan el marco subyacente a las pautas:

- Proporcionar múltiples formas de representación, el qué del aprendizaje.
- Proporcionar Múltiples formas de acción y Expresión, el cómo del aprendizaje.
- Proporcionar Múltiples formas de Implicación, el por qué del aprendizaje.

## 1.2 Objetivos del proyecto

El objetivo del portal YourMOOC4all es precisamente recoger información sobre la oferta de MOOCs en relación con su diseño inclusivo, el nivel de usabilidad y accesibilidad alcanzado y de su potencial integración y uso con tecnologías asistivas, de forma que sirva de guía y recomendación para estudiantes con discapacidad ajustándose a sus necesidades de adaptación.

El portal recoge la información de dos formas:

- Recogida automatizada (Harvesting) de la información general de los cursos: quien los imparte, donde se imparten, tiempo de duración, fecha de comienzo, requisitos previos, etc.
- Retroalimentación de los usuarios registrados en el portal, incluyendo las votaciones y los comentarios que ha recibido un curso por parte de sus participantes.

Este proyecto hace mucho énfasis en la recolección de la información sobre la experiencia de usuario percibida por otros usuarios registrados en el portal y que hayan realizado ese curso MOOC con anterioridad. Los usuarios que se registren en la plataforma también podrán aportar sus valoraciones de los cursos que vayan realizando de forma que sirva como guía para otros estudiantes. Los usuarios registrados también tienen la posibilidad de marcar los cursos que les puedan resultar de interés, de manera que el sistema se lo muestra al iniciar sesión y no necesitan repetir las búsquedas para localizarlos.

El sistema también es capaz de realizar un análisis exhaustivo (Harvesting) de las diferentes plataformas que ofrecen MOOCs en español de manera regular, hasta una vez al día, para que la oferta de cursos no se vaya quedando obsoleta y mantenerla actualizada. El administrador será el encargado de supervisar todo este trabajo autónomo que realiza la aplicación web.

El portal da, de momento, soporte en dos idiomas: español e inglés, pero la aplicación está diseñada para que se pueda ampliar la lista en un futuro con facilidad para el desarrollador.

### **1.3 Tecnologías utilizadas**

En este apartado se desglosan las tecnologías utilizadas a lo largo del proyecto, mayoritariamente de código abierto y servicios gratuitos en la nube, empezando por el lenguaje utilizado y terminando por el editor de texto, que será Sublime Text. Este es un potente y ligero editor de código multiplataforma.

Para el alojamiento en la web en desarrollo se ha empleado la plataforma Heroku. Sin embargo, se podría implementar un servidor web propio, este se montaría con un sistema operativo Ubuntu Server en su versión 17.4, con Apache para servir las páginas estáticas y Passenger para hacer de servidor de aplicaciones, procesar las páginas dinámicas escritas en Ruby convirtiendo estas en páginas estáticas.

Para obtener la información de los proveedores MOOC se ha implementado una pasarela mediante el estándar Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) que define el XML (formato, etiquetas, ...) del contenido que se puede recopilar.

### 1.3.1 Ruby, Ruby On Rails y sus librerías

Ruby Versión Manager es una herramienta en línea de comandos que permite de manera fácil: instalar, gestionar y trabajar con múltiples entornos de Ruby y sus diferentes gemas instaladas en los diferentes entornos <sup>1</sup>. Esta herramienta está pensada para trabajar en un mismo sistema operativo con diferentes proyectos escritos con diferentes versiones de Ruby. Esto que en principio parece que no ocurrirá nunca, en el mundo del software es algo normal, ya que las versiones del software van evolucionando en el tiempo.

El lenguaje empleado para la lógica de negocio es Ruby [8] en su versión 2.3.6, que al ser un lenguaje interpretado se adapta perfectamente a los desarrollos web. Ruby es un lenguaje de programación dinámico y de código abierto enfocado en la simplicidad y la productividad. Su sintaxis es fácil de leer y escribir, lo que facilita su aprendizaje <sup>2</sup>. Es un lenguaje interpretado Orientado a Objetos de propósito general y multiplataforma, esto lo convierte en un lenguaje poderoso y versátil para crear código sencillo y eficiente. Creado para resolver problemas generales en el área de la computación, sobre todo scripts de mantenimiento y tareas repetitivas, pero su verdadera popularización llegó de la mano del Framework Ruby On Rails. Fue creado por Yukihiro Matsumoto y disfruta de una comunidad muy activa a nivel mundial que continua en expansión. Cuenta con una amplia gama de paquetes llamados gemas, lo que vendría a ser las librerías de terceros, que agilizan el trabajo. Combina la sintaxis de otros lenguajes de programación como Python y Perl, con características de programación orientada a objetos. También comparte funcionalidades con lenguajes como Lisp, Lua, Dylan y CLU.

Se ha empleado Ruby On Rails en su versión 4.2.10, para el Framework utilizado en el back-end de la aplicación web. Facilita enormemente la construcción de grandes aplicaciones en el servidor. Fue creado por David Heinemeier Hansson [9] [10] [11] [12]. Es un Framework que domina a la perfección el paradigma del Modelo-Vista-Controlador, el cual permite tener de forma totalmente separada la codificación para la base de datos, la codificación del modelo de negocio y la codificación de las vistas que hay que generar.

Este Framework se basa en dos filosofías básicas:

- Don't Repeat yourself, lo que ya está hecho no tiene porqué volver a hacerse
- Usar las convenciones sobre las configuraciones, de esta manera es muy fácil generar un código útil sin ninguna configuración previa.

También incluye el concepto de migraciones, que permite definir o modificar la estructura de la base de datos desde Rails, y además de una forma independiente del motor de base de datos elegido. ActiveRecord es un Object Relational Mapping o en castellano Mapeo de Objeto Relacional, su uso evita tener que escribir código SQL para interactuar con la base de datos y

---

<sup>1</sup> <https://rvm.io/rvm/install>

<sup>2</sup> <https://www.ruby-lang.org/es/>

elimina las tareas que normalmente se realizan a mano. Uno de los aspectos más importantes es que está basado en convenciones en lugar de configuraciones, por lo que no es necesario utilizar archivos de configuración para acceder a las tablas de la base de datos. Otra de las ventajas de utilizar ActiveRecord es la posibilidad de llamar a métodos de un objeto de datos en vez de a la base de datos, así se pueden crear métodos en el objeto para tener campos calculados en tiempo real. En este proyecto se ha usado ActiveRecord como herramienta ORM para acceder a la base de datos y manipular el contenido de las tablas, esto ha resultado muy útil y productivo ya que se puede acceder a los datos de la base de datos relacional convertidos en objetos.

### 1.3.2 HTML, CSS y JavaScript

Como lenguajes de maquetación se ha empleado HTML5 y CSS, al que se debe sumar JavaScript para la parte dinámica de la misma. HTML es un lenguaje de marcado que se utiliza para el desarrollo de páginas de Internet. Las siglas que corresponde a HyperText Markup Language, es decir, Lenguaje de Marcas de Hipertexto. Sirve como andamiaje para poder marcar el contenido que aparece en la página generada, como por ejemplo texto, fotografías, animaciones, etc. Una vez marcado el documento se pueden implementar el resto de las tecnologías que se usan en el desarrollo de la aplicación web [13] [14]. Fue desarrollado por la Organización Europea de Investigación Nuclear (CERN) con la finalidad de desarrollar un sistema de almacenamiento donde las cosas no se perdieran, que pudieran ser conectadas a través de hipervínculos. Ted Nelson acuñó el término hipervínculo, ideando una estructura que se encontraba conectada de forma electrónica y que más tarde permitiría la creación de la World Wide Web.

CSS es un lenguaje de estilo que define la presentación de los documentos HTML. Gestiona todo lo referente a fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento avanzado y muchos más [14]. Las hojas de estilos aparecieron poco después que el lenguaje de etiquetas SGML. Desde la creación de SGML, se observó la necesidad de definir un mecanismo que permitiera aplicar de forma consistente diferentes estilos a los documentos electrónicos. Lie y Bos se unieron para definir un nuevo lenguaje que tomaba lo mejor de cada propuesta y lo llamaron CSS (Cascading Style Sheets).

JavaScript es un lenguaje de programación que surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario y la realización de automatismos sencillos [14]. Nació como un “lenguaje de scripting” del lado del cliente y aportan el dinamismo necesario para el manejo del DOM, modificándolo, refrescándolo, supervisando las acciones del usuario e interactuando con este último. También aportan comunicación con el servidor a través de socket o solicitan información para modificar la página (sin necesidad de utilizar llamadas POST o GET). Además, en los últimos años JavaScript se está convirtiendo también en el lenguaje “integrador”. Se encuentra en muchos ámbitos, ya no solo en Internet y la Web, también es nativo en sistemas operativos para ordenadores y dispositivos, del lado del servidor y del cliente.

A continuación, se van a describir las gemas más importantes, pero no las únicas, que se han utilizado en el desarrollo de este proyecto.

Devise es una gema que permite que los usuarios creen cuentas en la página web, modifiquen sus perfiles, inicien o cierren su sesión, que se envíen los recordatorios de sus contraseñas, etc. De forma segura y fácil de configurar<sup>3</sup>. CanCan es el complemento perfecto para la gema Devise, ya que a través de un fichero llamado Ability, se controla lo que puede o no puede hacer un usuario de la aplicación.

Tiene 10 módulos de funcionamiento:

- Database Authenticatable, encripta la contraseña y se autentica al usuario vía solicitud POST o HTTP.
- Omniauthable, permite registrar y acceder utilizando Facebook, Twitter y demás aplicaciones que permitan la autenticación multiproveedor.
- Confirmable, envía mensajes de correo electrónico con instrucciones para confirmar y verifica la cuenta de usuario.
- Recoverable, sirve para resetear la contraseña, envía instrucciones al usuario para hacerlo.
- Registerable, permite al usuario registrarse en la aplicación y maneja datos de usuarios, para editar o incluso eliminar la cuenta.
- Rememberable, permite grabar en una cookie y dejar guardados los datos del usuario.
- Trackable, se puede ver la información sobre las conexiones que se han tenido, como, por ejemplo: el número de entradas, tiempo y dirección IP.
- Timeoutable, las sesiones que no están activas durante un período de tiempo se cierran.
- Validatable, validaciones de correo electrónico y contraseña de forma personalizada.
- Lockable, bloquea una cuenta después de un número determinado de intentos.

La gema Nokogiri es una biblioteca que sirve para analizar las páginas HTML, es un parseador de archivos HTML, XML y SAX. Entre las muchas características de Nokogiri, la que más destaca es la capacidad de buscar documentos a través de XPath o selectores de CSS3. Nokogiri analiza y busca en los XML / HTML muy rápidamente. Es muy útil a la hora de parsear contenido de forma rápida y sobre todo segura<sup>4</sup>. Con esta gema se realiza el Harvesting sobre los sitios web que ofrecen cursos MOOC para llenar la tabla Cursos de la base de datos. La gema debe estar previamente instalada en el sistema y una vez instalada ya se puede incluir en el algoritmo de adquisición de datos con simple require. Otra gema llamada open-uri, que permite manejar protocolos como http, https y ftp, para tratar los documentos que se analizan como si estuvieran en local. Antes de su uso se le indica a la gema Nokogiri que lo que se va a leer es un HTML. En la variable se obtiene toda la página web, por lo que con la ayuda

---

<sup>3</sup> <https://github.com/RailsApps/rails3-bootstrap-devise-canCan>

<sup>4</sup> <http://ruby.bastardsbook.com/chapters/html-parsing/>

de los métodos `css('identificador')` y `at_css('identificador')` se puede obtener el texto que contengan las clases o identificadores correspondientes. Con la primera de ellas se va a obtener el primer texto que se encuentre en la primera etiqueta, y con el segundo método se obtienen todos los textos del documento.

Se muestran una serie de selectores CSS para la gema Nokogiri:

- El elemento `<Title>`: `doc.css('title')`
- Todos los elementos `<li>`: `doc.css('li')`
- El texto del primer elemento `<li>`: `doc.css('li')[0].text`
- La url del segundo elemento `<li>`: `doc.css('li')[1]['href']`
- Los elementos `<li>` con la propiedad `data-category` igual a news: `doc.css("li[data-category='news'])"`
- El elemento `<div>` con el ID "funstuff": `doc.css('div#funstuff')[0]`

Para paginar los resultados obtenidos en las consultas a las tablas de la base de datos se utiliza la gema Will\_paginate. El uso de esta gema es muy sencillo y permite paginar cualquier colección de elementos, en este caso los obtenidos desde el modelo<sup>5</sup>. Para ello hay que insertar código en la ubicación que contenga la relación de páginas que se ha generado y los enlaces para poder avanzar o retroceder la paginación.

El uso de la gema Paperclip ha sido finalmente desestimada puesto que no está soportada en Heroku. Al comienzo de la implementación del proyecto se utilizó para guardar las fotos de los cursos que se iban capturando y almacenando en las bases de datos<sup>6</sup>. Paperclip es una librería para adjuntar ficheros en ActiveRecord y así poder tratar el fichero como un campo más de la base de datos. El fichero no se guarda dentro de la base de datos, sino en el directorio public de la aplicación web. Lo que realmente se guarda en la base de datos es una referencia al directorio donde está ubicado el fichero y sus miniaturas. También cuenta con validaciones para el tipo de fichero y herramientas para convertir la imagen en diferentes tamaños.

Delayed\_job es una gema que permite ejecutar tareas que necesitan mucho tiempo de procesado, como por ejemplo el Harvesting, la adquisición de información desde las plataformas web de cursos MOOC. Con esta gema se puede ejecutar la tarea en segundo plano, con lo que se logra que una tarea de larga duración no provoque en el navegador el temido mensaje “el servidor no responde”<sup>7</sup>. Para que las tareas que se vayan ejecutando en segundo plano, el servidor de aplicaciones debe tener un demonio en ejecución, como el que

---

<sup>5</sup> [https://rubyinrails.com/2014/03/27/pagination-in-rails-using-will\\_paginate-gem/](https://rubyinrails.com/2014/03/27/pagination-in-rails-using-will_paginate-gem/)

<sup>6</sup> <https://code.tutsplus.com/es/tutorials/rails-image-upload-using-paperclip-in-a-rails-application--cms-25974>

<sup>7</sup> [https://github.com/collectiveidea/delayed\\_job](https://github.com/collectiveidea/delayed_job)

ofrece de forma gratuita Heroku. Sin este proceso en ejecución la gema `Delayed_job` almacenaría las tareas, pero nunca las llevaría a cabo.

### 1.3.3 Servidores de bases de datos

Este proyecto necesita almacenar información tanto de los usuarios de la aplicación web, como de los cursos adquiridos durante el análisis de los sitios oferentes, etc. Por este motivo la aplicación necesita un servidor de base de datos. Son necesarios dos servidores de bases de datos diferentes para este proyecto, uno será MySQL<sup>8</sup>, para la parte de desarrollo instalada en local. El servidor de bases de datos del alojamiento web será PostgreSQL por imposición de Heroku. Ruby On Rails puede usar diferentes servidores de bases de datos en los tres ambientes definidos: desarrollo, test y producción.

El sistema de base de datos operacional MySQL es hoy en día uno de los más importantes en lo que hace al diseño y programación de base de datos de tipo relacional. Cuenta con millones de aplicaciones y aparece en el mundo informático como una de las más utilizadas por usuarios del medio. El programa MySQL se usa como servidor a través del cual pueden conectarse múltiples usuarios y utilizarlo al mismo tiempo. Una de las características más interesantes de MySQL es que permite recurrir a bases de datos multiusuario a través de la web y en diferentes lenguajes de programación que se adaptan a diferentes necesidades y requerimientos. Por otro lado, MySQL es conocida por desarrollar alta velocidad en la búsqueda de datos e información, a diferencia de sistemas anteriores. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad. Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. MySQL es software de código abierto. Cualquier persona puede descargar el código fuente de MySQL y usarlo de forma libre y gratuita y ajustarlo a sus necesidades. MySQL usa el GPL (GNU General Public License) para definir los límites de utilización del software en diferentes situaciones.

Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL.
- Disponibilidad en gran cantidad de plataformas y sistemas.

---

<sup>8</sup> <https://dev.mysql.com/doc/refman/5.7/en/>

- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferente velocidad.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

PostgreSQL se ha utilizado en la parte de producción, ya que era un requisito ineludible en el alojamiento web en Heroku. Con Ruby On Rails no se ha tenido ningún problema a la hora de utilizar dos servidores de bases de datos diferentes, porque Ruby On Rails dispone de tres configuraciones diferentes, una para cada ambiente. A continuación, se muestra el fichero databases.yml de nuestro proyecto<sup>9</sup>. PostgreSQL es una de las opciones más interesantes en bases de datos relacionales open-source. Michael Stonebraker inició el proyecto bajo el nombre Post Ingres a mediados de los 80's con la idea de solucionar problemas existentes en las bases de datos en esa época. MySQL fue por mucho tiempo el motor más popular; pero hoy es propiedad de Oracle y esto limita su evolución. Es gratuito y libre, además de que hoy ofrece una gran cantidad de opciones avanzadas. De hecho, es considerado el motor de base de datos más avanzado en la actualidad. Una característica interesante de PostgreSQL es el control de concurrencias multiversión; o MVCC por sus siglas en inglés. Este método agrega una imagen del estado de la base de datos a cada transacción. Esto permite hacer transacciones eventualmente consistentes, ofreciendo grandes ventajas en el rendimiento. En PostgreSQL no se requiere usar bloqueos de lectura al realizar una transacción lo que brinda una mayor escalabilidad. También PostgreSQL tiene Hot-Standby. Este permite que los clientes hagan búsquedas (sólo de lectura) en los servidores mientras están en modo de recuperación o espera. Así se pueden hacer tareas de mantenimiento o recuperación sin bloquear completamente el sistema. PostgreSQL aporta mucha flexibilidad a los proyectos. Por ejemplo, permite definir funciones personalizadas por medio de varios lenguajes. Algunos son: PL/pgSQL, PL/Tcl, PL/Perl, PL/Python, PL/PHP, PL/Ruby y PL/Java.

### 1.3.4 jQuery, Bootstrap y Chartkick

JQuery esta es una biblioteca de JavaScript para manejar de una manera fácil el DOM y los eventos de la navegación. Por ejemplo, cuando se pulsa sobre el botón borrar un curso, se captura el evento desde JavaScript, se cambia el literal del botón, y si el botón se vuelve a pulsar antes de cuatro segundos, se procede a borrar el curso<sup>10</sup>.

Por defecto Ruby On Rails utiliza CoffeeScript, este es un lenguaje que convierte automáticamente a JavaScript. Con cada controlador se crea un archivo en app/assets/JavaScript con el nombre del controlador y la extensión .coffee. En este proyecto

---

<sup>9</sup> <https://www.postgresql.org/docs/10/static/index.html>

<sup>10</sup> <http://api.jquery.com/>

no se hace uso de CoffeeScript, sino que se cambia la extensión de archivo a .js para utilizar JavaScript de forma nativa.

Para aportar interoperabilidad, el Framework Bootstrap es el más adecuado, el cual va a permitir que el diseño web sea adaptativo a los diferentes dispositivos actuales. Bootstrap es un Framework CSS y JavaScript diseñado para la creación de interfaces Responsive Design. Ofrece un amplio abanico de componentes y funciones para que los usuarios puedan crear un sitio web de forma sencilla y muy productiva<sup>11</sup>. Es una herramienta muy popular entre los desarrolladores web. Una de las grandes ventajas de este Framework es adaptación a cualquier tipo de dispositivo electrónico. Algo con mucha importancia hoy en día, ya que cada vez hay más dispositivos conectados a Internet con diferentes tamaños de pantalla.

Contiene plantillas de diseño con tipografías, formularios, botones, cuadros de diálogo, menús de navegación y otros elementos basados en HTML, CSS y JavaScript. En resumen, se ocupa del aspecto de la parte del cliente o front-end. Es posible ajustar el Framework de una forma limitada, a través de una hoja de estilo de configuración. Los cambios más profundos son posibles mediante las declaraciones LESS. El uso del lenguaje de hojas de estilo LESS permite el uso de variables, funciones y operadores y selectores anidados. El sistema de rejilla está pensado para adaptar el contenido del sitio web al tamaño de pantalla de los diferentes dispositivos de forma automática. Para ello hay que poner los componentes de la página dentro de columnas, 12 como máximo, que se irá ajustando de forma dinámica.

Pantalla	Prefijo de la clase	Ancho del contenedor
Tamaño extra pequeño Teléfonos (<768px)	.col-xs-	Ninguno (automático)
Tamaño pequeño Tabletas (>=768px)	.col-sm-	750px
Tamaño medio Escritorios (>=992px)	.col-md-	970px
Tamaño grande Escritorios (>= 1200px)	.col-lg-	1170px

Tabla 1 - Bootstrap

Chartkick es una biblioteca de gráficos para Ruby On Rails, que consiste en una adaptación de Google charts, la cual permite crear fácilmente gráficos muy espectaculares, además es compatible con la mayoría de los navegadores actuales. Se ha aplicado a las evaluaciones recibidas según las plataformas, pero la biblioteca cuenta con muchos tipos de gráficos<sup>12</sup>.

---

<sup>11</sup> <http://getbootstrap.com/2.3.2/components.html>

<sup>12</sup> <https://www.chartkick.com/>

## 1.4 Estructura de la memoria

En los siguientes apartados se puede ver en detalle el desarrollo del software de este proyecto. Comenzando por los análisis de requisitos como: de usuarios, de registro, de interfaz, de Harvesting y de resultados. También se ha incorporado el modelo de dominio, para proporcionar una idea del alcance del proyecto y un modelo conceptual para ver la relación que existe entre los diferentes escenarios que va a representar el sistema.

Gran parte del análisis se ha realizado a través de los casos de uso, su diagrama de casos de uso, sus diagramas de secuencia y los escenarios alternativos. Toda esta información se ha realizado en un lenguaje claro y cercano al usuario del sistema, dejando claro cuál es el papel de cada actor en cada caso de uso. También se ha recabado información sobre los requisitos funcionales, no funcionales y de seguridad.

En el apartado de diseño se expone la arquitectura del sistema, explicando someramente el Modelo Vista Controlador, el servicio REST y el modelo Cliente-Servidor. También se aporta la guía de estilo y diseño enfocado a la usabilidad y la accesibilidad. Seguidamente se puede ver una muestra de lo que es la interfaz de usuario y un diagrama de navegación. También se utilizaron elementos y modelos obtenidos de análisis para transformarlos en mecanismos que puedan ser utilizados en un entorno web. Se diseñaron todos los niveles de los que consta la aplicación: nivel de presentación, lógico y persistencia.

En el apartado de implementación se detallan cada uno de los componentes, sus principales características, particularidades y problemas específicos que han debido ser resueltos para la finalización exitosa del proyecto.

Durante la fase de pruebas se realizaron comprobaciones del correcto funcionamiento del producto desarrollado mediante una serie de pruebas exhaustivas. Finalmente se han incluido un apartado que explica el control de versiones y otro más para el alojamiento de la aplicación en la nube.

## 2. Análisis

El análisis global de los requisitos de una aplicación web es un proceso de conceptualización y formulación de las distintas características y procesos funcionales que van a ser ofrecidos a los usuarios. Es una parte fundamental del proceso de desarrollo de una aplicación y se pueden distinguir:

- Requisitos del cliente: documentan los deseos y necesidades de los clientes y se expresan en lenguaje claro para él.
- Requisitos detallados: Determina los requisitos de manera específica y estructurada y están destinados específicamente hacia los desarrolladores.

A continuación, se exponen los requerimientos detallados de forma clara y sistemática desde el punto de vista del desarrollador.

### 2.1 Análisis de requisitos

El objetivo general desarrollado en este proyecto ha sido una aplicación web para la búsqueda en cursos MOOC en diferentes plataformas y que ofrece una búsqueda parametrizada con resultados en forma de ranking.

#### 2.1.1 Requisitos de tipos de usuarios o roles

Se distinguen tres roles distintos para los usuarios de la misma: el primero será un “usuario” sin ningún tipo de registro en la aplicación, el segundo será un “usuario registrado”, este usuario especial debe contar con una cuenta en nuestro sistema, y por último está el usuario “administrador”, el cual es un usuario registrado con plena capacidad de actuación en el sitio web.

El usuario podrá hacer búsquedas en el sistema, consultar la puntuación obtenida por los usuarios registrados, ver los detalles de un curso en particular y abrir una nueva pestaña en el navegador con la dirección del curso en el que podrá hacer el registro en el mismo.

El usuario registrado podrá realizar todo lo que hace un usuario estándar, más lo siguiente: podrá valorar un curso que esté realizando, que haya abandonado o que lo haya terminado. También podrá marcar un curso determinado como de su interés, esto implica que cada vez que un usuario registrado con cursos marcados llegue a su página de inicio, el sistema le mostrará todos los cursos que este usuario haya marcado. Por supuesto una vez que un curso ya no sea de su interés podrá ser desmarcado y el sistema ya no se lo mostrará más.

El administrador del sistema deberá supervisar todos los procesos que realiza el sistema de forma autónoma. También deberá gestionar las cuentas de usuario, estas cuentas son creadas por los propios usuarios, pero deberán ser sometidas a la supervisión del administrador, este podrá cambiar la contraseña de un usuario, ver su actividad, sus votaciones e incluso borrar su cuenta.

El administrador también deberá supervisar la información de los cursos que el sistema obtiene de forma automática, corrigiendo los fallos que haya cometido el sistema de adquisición autónoma, o Harvesting. El administrador deberá marcar como oculto los cursos que estime, para que no puedan ser encontrados por los usuarios en sus búsquedas. Otra tarea de la que se tiene que hacer cargo el administrador será el mantenimiento de las tablas: Plataformas, Instituciones, Idiomas, Audios, Subtítulos y Transcripciones.

#### 2.1.2 Requisitos de registro

Al inicio, el usuario debe registrarse en el sistema aportando su correo electrónico y una contraseña válida, esto significa que tal contraseña debe tener un mínimo de 6 caracteres, también debe repetir tal contraseña para verificar que inicialmente estaba bien escrita. Una vez que finaliza el registro, se muestra al usuario registrado una página de inicio donde podrá ver sus cursos marcados como de interés, editar sus datos de registro, cambiar su contraseña, aplicar valoraciones a los cursos o salir del sistema.

#### 2.1.3 Requisitos de interfaz de búsquedas

La búsqueda será parametrizable en cuanto a los campos en los que se va a realizar la comparación con la cadena introducida por el usuario, y en cuanto al orden en el que se obtendrán los resultados de la búsqueda. Estos parámetros se guardarán en la sesión del navegador para que el usuario no deba repetir de forma continua la selección de los parámetros.

#### 2.1.4 Requisitos de Harvesting

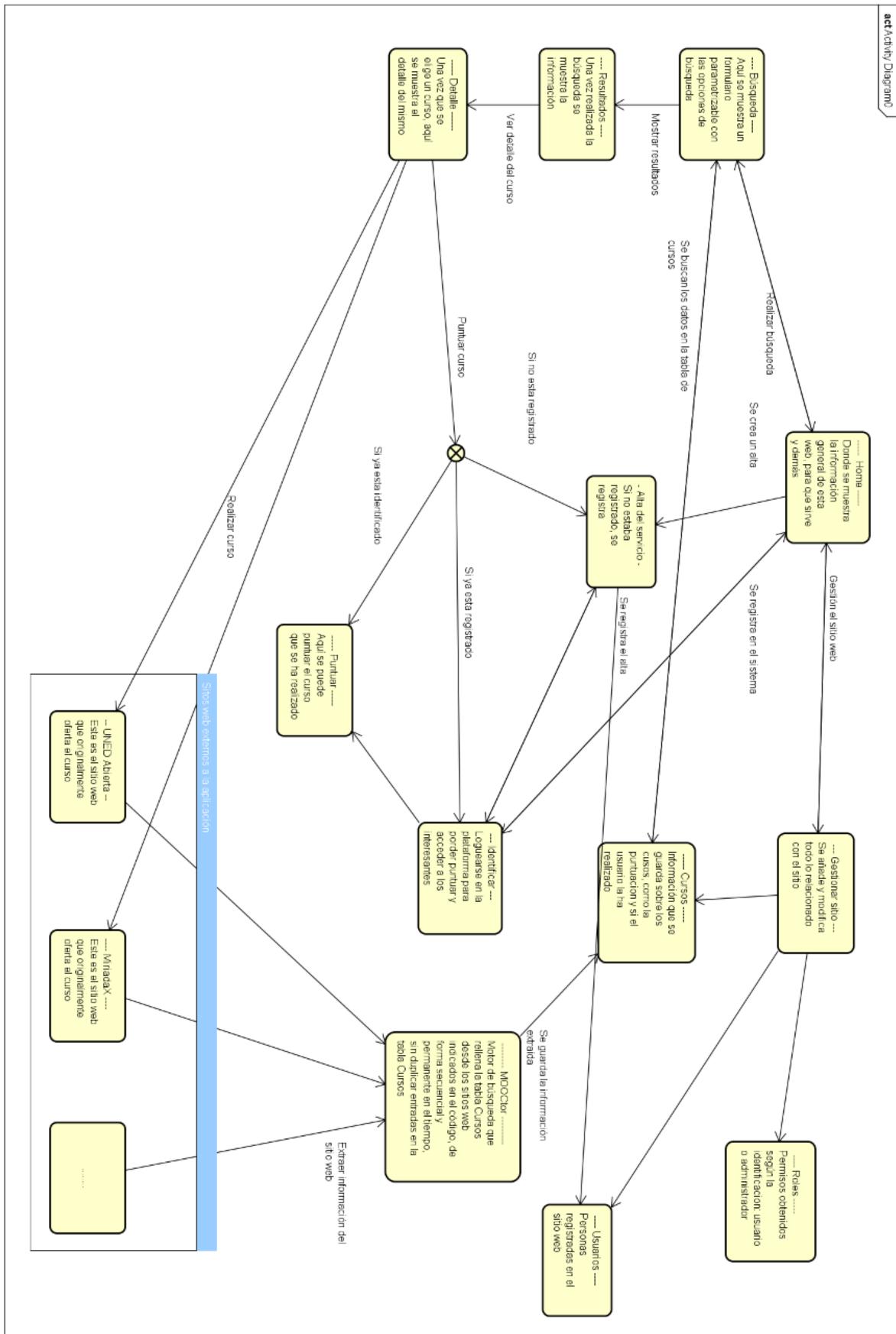
Para poder realizar las búsquedas mencionadas en el apartado anterior, el sistema debe haber realizado con anterioridad un recopilación y análisis de las plataformas oferentes de cursos a través de un Harvesting automatizado. Esta será una tarea autónoma y repetida una vez al día sin la intervención de ningún administrador, que también admite que un administrador pueda activar un análisis en un momento determinado. El Harvesting hará una petición al servidor de la plataforma obteniendo la página web con los cursos que ella ofrece, acto seguido, el sistema deberá hacer un recorrido por todos los cursos disponibles e identificar la información contenida en diferentes partes del código HTML, seleccionar estas partes de información para después introducirlas en los campos definidos en la tabla Cursos.

### 2.1.5 Requisitos de interfaz de resultados

Las votaciones que han realizado los usuarios registrados deben ser almacenadas en una tabla vinculada a los cursos, incluso a los cursos que son de diferentes ediciones. Esto significa que, si un curso en su primera edición obtiene una puntuación de 4 puntos sobre 5, y el año que viene se crea una segunda edición de este mismo curso, el sistema una vez que ha sido informado por el administrador, deberá mostrar esa puntuación de 4 puntos en la nueva edición del curso, aún que este último todavía no haya recibido ninguna valoración.

## 2.2 Modelo del dominio

En el siguiente modelo se va a capturar y representar el entendimiento adquirido sobre el alcance del sistema, como un mapa conceptual de lo que será la aplicación una vez terminada. Es un buen punto de partida para extraer casos de uso y así tener una aproximación escalonada hacia el entendimiento del problema. Al seccionar el modelo del dominio en casos de uso se puede abordar la problemática de la aplicación web en trabajos más pequeños.



*Figura 1 - Modelo de dominio*

## 2.3 Diagrama de casos de uso

Los diagramas de casos de uso son un método alternativo y complementario a los diagramas de contexto para especificar los requisitos de una aplicación software. Muestran los tipos básicos de interacción entre el sistema y los elementos del entorno que operan con él. Los casos de uso capturan una vista general de la funcionalidad del sistema que puede ser interpretado por personas no técnicas como son los usuarios y los expertos en dominio y forman un método de descomposición de la funcionalidad del sistema en elementos de funcionalidad más básica.

Para realizar el diagrama de casos de uso, primero hay que identificar los actores que participarán en dichos casos de uso, que son los siguientes:

- Usuario
- Usuario registrado
- Administrador

Estos tres actores identificados tienen roles que van de menos a más, por ejemplo, el usuario sin registrar (a partir de ahora referenciado como usuarios a secas) solo puede hacer búsquedas en el sistema y ver los detalles del curso que está buscando, como la puntuación, el esfuerzo estimado, la fecha para la próxima edición, la información y temática. También puede hacer clic sobre el enlace al curso en la plataforma que lo ofrece.

El usuario registrado puede realizar acciones como votar los cursos que ha realizado y añadir a su lista personal los cursos que le parecen interesantes, además de las acciones que ya se han descrito para el usuario sin registrar.

El administrador del sistema, además de lo anterior, puede lanzar el procedimiento que adquiere y analiza la información en la web sobre los múltiples cursos que ofrecen las plataformas de formación online. También puede administrar estos cursos, los usuarios, los idiomas, los audios de los cursos, los subtítulos y las transcripciones.

Ahora que están definidos los actores, se va a desglosar los casos de uso para el sistema:

- Buscar curso en el sistema
- Cambiar idioma del sistema
- Registrarse en el sistema
- Identificarse en el sistema
- Recuperar contraseña

- Votar un curso
- Seleccionar un curso como interesante
- Adquirir información en las diferentes plataformas
- Gestionar los cursos
- Gestionar las plataformas
- Gestionar las Instituciones
- Gestionar los idiomas
- Gestionar los usuarios

Una vez que se han definido los casos de uso y los actores, se ha procedido a definir las relaciones entre ellos, es decir quien realiza qué acción.

uc

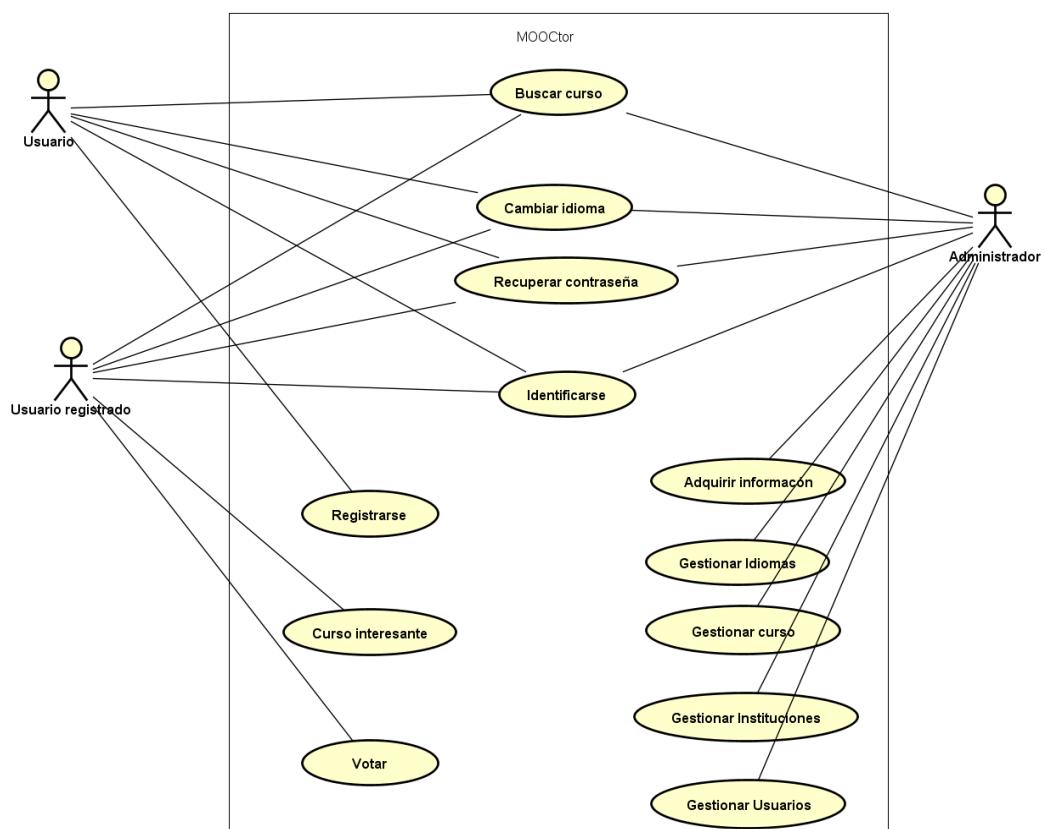


Figura 2 -Diagrama de casos de uso

## 2.4 Listado de casos de uso

Los casos de uso son las descripciones de las actividades que deben realizarse para llevar a cabo algún proceso de comunicación o comportamiento mediante su interacción entre el sistema y sus actores.

### 2.4.1 Caso de uso: Buscar curso en el sistema

Este caso de uso es compartido por los tres actores principales, con la única diferencia del aspecto visual, sobre todo si el actor es el administrador, pero en esencia el caso de uso es el mismo. Un actor necesita hacer una búsqueda y el sistema se lo proporciona.

- **Actor principal:** Usuario
- **Precondiciones:** El usuario tiene abierto un navegador con conexión a internet y dispone de la dirección de nuestro sitio web, bien porque ha buscado a través de Google o bien porque ya conocía la URL en internet.
- **Postcondiciones:** El usuario encuentra el curso que estaba buscando y accede al registro del curso buscado en el sitio web oferente del mismo.
- **Escenario principal de éxito:**

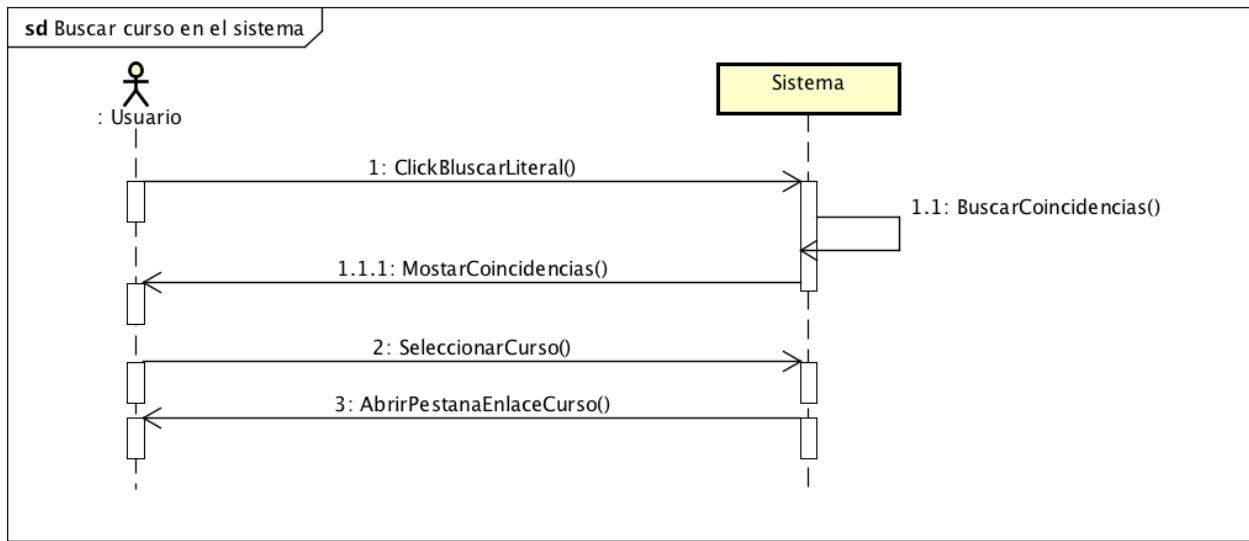
Acción del actor (Usuario)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El usuario escribe un literal en la etiqueta input, selecciona un orden para el resultado de la búsqueda y un campo sobre el que buscar. Presiona el botón buscar.	4 - El sistema hace una búsqueda de los cursos que coinciden con los criterios solicitados, devuelve una colección de cursos con el orden coincidente con el solicitado.
5 - El usuario navega por las diferentes páginas con los resultados obtenidos y selecciona el curso que es de su interés.	6 - El sistema muestra información detallada sobre el curso seleccionado.
7 - El usuario lee la información sobre el curso y decide que va a inscribirse en él, haciendo clic en el botón de enlace al curso.	8 - El sistema abre una pestaña nueva en el navegador con la dirección del curso para que el usuario pueda llevar a cabo la inscripción si lo estima oportuno.

Tabla 2 - caso de uso 1

- **Flujo alternativo:**

4 – El sistema no encuentra ningún curso que contenga el literal que el usuario ha proporcionado, muestra un mensaje con el siguiente literal “No se han encontrado coincidencias en nuestros archivos”.

### DSS: Buscar curso en el sistema



powered by Astah

Figura 3 - caso de uso 1

#### 2.4.2 Caso de uso: Cambiar idioma del sistema

Nuestro sistema pretende dar soporte para varios idiomas, actualmente solo se proporcionan dos idiomas: castellano e inglés. El diseño final está definido para que se admitan más idiomas en el futuro de una forma muy fácil, eso significa que se podrán añadir más idiomas sin que se deba modificar ninguna línea de código para ello.

- **Actor principal:** Usuario
- **Precondiciones:** El usuario tiene abierto un navegador con conexión a internet y está posicionado en la página de inicio para los usuarios que no están registrados en el sistema.
- **Postcondiciones:** El usuario cambia el idioma de los literales para nuestro sistema.
- **Escenario principal de éxito:**

Acción del actor (Usuario)	Responsabilidad del sistema
1 - El usuario desde la página de inicio, o cualquier otra que tenga disponible la opción de elegir el idioma que prefiere.	2 - El sistema muestra la página en el idioma por defecto, que en nuestro caso es el castellano.
3 - El usuario pulsa sobre el enlace con la palabra 'inglés' porque prefiere el lenguaje de Shakespeare al de Cervantes.	4 - El sistema vuelve a cargar la página, pero esta vez con el idioma inglés en los literales.

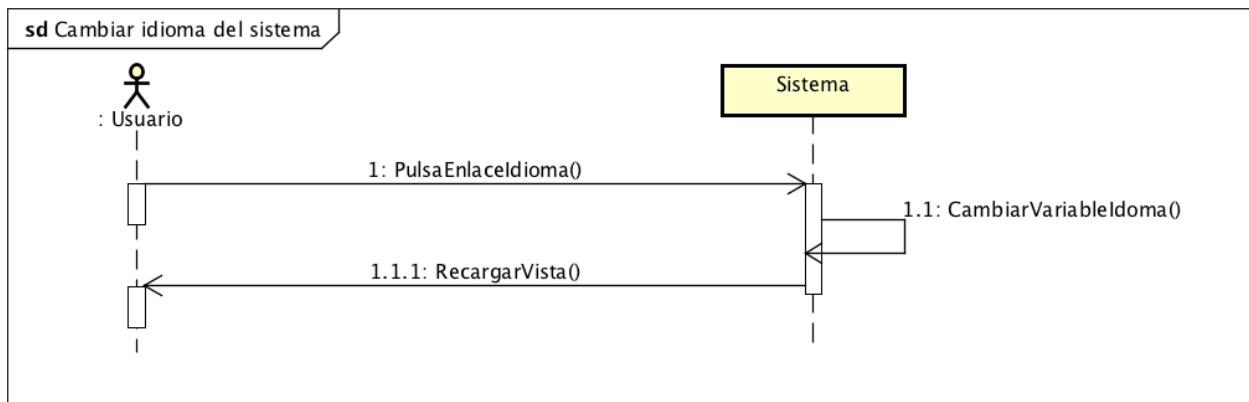
Tabla 3 - caso de uso 2

- **Flujo alternativo:**

1 - El usuario selecciona un idioma que ya está marcado.

2 - El sistema no realiza ninguna acción.

DSS: Cambiar idioma del sistema



powered by Astah

Figura 4 - caso de uso 2

#### 2.4.3 Caso de uso: Registrarse en el sistema

Para poder evaluar un curso o marcarlo como de interés, hay que estar registrado en el sistema, por ello el registro está automatizado. Este enfoque repercute positivamente en el administrador, ya que es el propio usuario interesado quien se da de alta de forma autónoma.

- **Actor principal:** Usuario
- **Precondiciones:** El usuario está posicionado en la página de inicio para los usuarios que no están registrados en el sistema.
- **Postcondiciones:** El usuario se da de alta en el servicio, se identifica en el sistema y termina visualizando la página inicial de los usuarios registrados.
- **Escenario principal de éxito:**

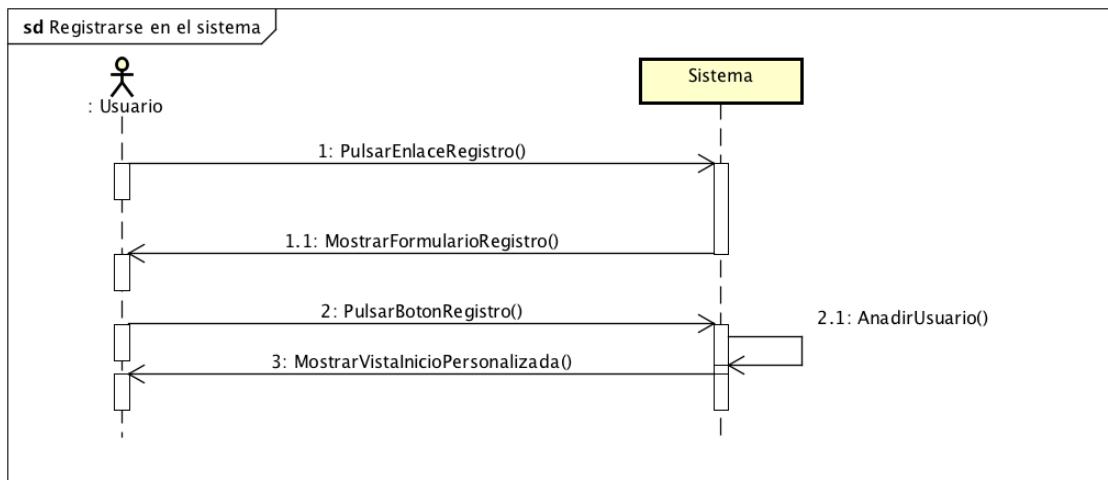
Acción del actor (Usuario)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El usuario pulsa sobre el enlace con la palabra Registrarse.	4 - El sistema muestra un pequeño formulario para que el usuario introduzca su email y su contraseña.
5 - El usuario rellena sus datos y pulsa el botón con el literal Registrarse.	6 - El sistema da de alta al usuario con un rol de usuario para que pueda votar y marcar cursos como interesantes.

Tabla 4 - caso de uso 3

- **Flujo alternativo:**

6 – El sistema detecta que el usuario ya se encuentra registrado, la dirección de correo electrónica no es válida o la contraseña es demasiado corta, es sistema muestra un mensaje indicando el problema y vuelve a solicitar los datos para el registro del nuevo usuario.

#### DSS: Registrarse en el sistema



powered by Astah

Figura 5 - caso de uso 3

#### 2.4.4 Caso de uso: Identificarse en el sistema

Una vez que ya existen usuarios registrados en el sistema hay que proporcionar un mecanismo para la validación de esos usuarios, el cual no pedirá de nuevo las credenciales para permitir la entrada en el sistema.

- **Actor principal:** Usuario registrado
- **Precondiciones:** El usuario está posicionado en la página de inicio para los usuarios que no están registrados en el sistema.
- **Postcondiciones:** El usuario se identifica en el sistema y visualiza la página inicial de los usuarios registrados, en la cual se encuentran los cursos marcados como interesantes para él.
- **Escenario principal de éxito:**

Acción del actor (Usuario registrado)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El usuario pulsa sobre el enlace con la palabra Identificarse.	4 - El sistema muestra un pequeño formulario para que el usuario introduzca su email y su contraseña.

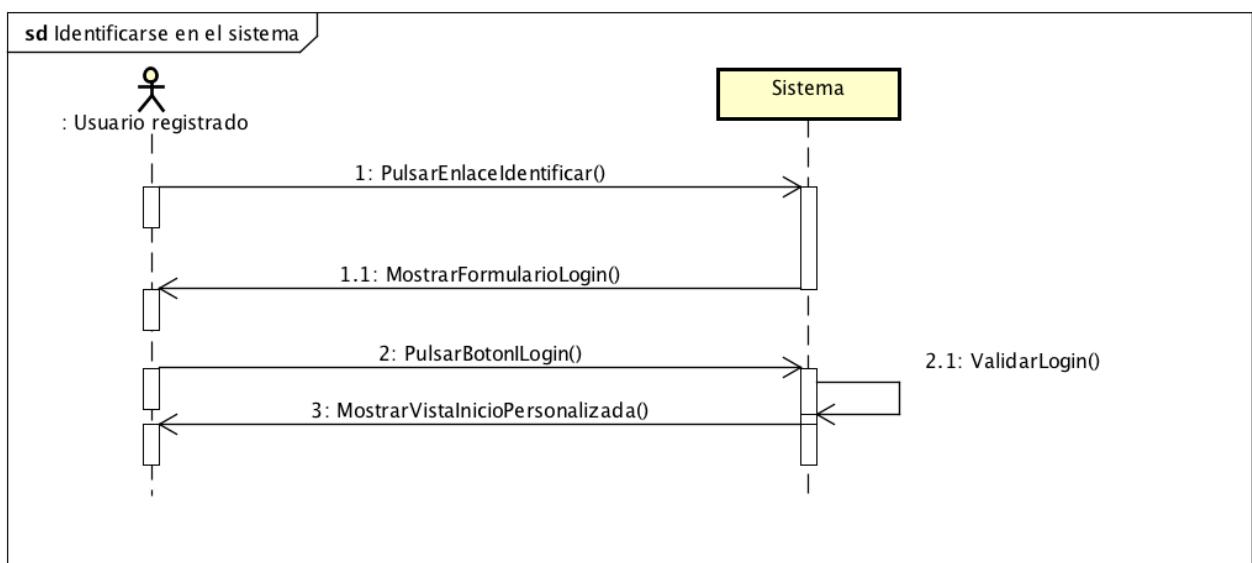
5 - El usuario rellena sus datos y pulsa el botón con el literal Identificarse.	6 - El sistema registra la entrada del usuario y le muestra la página de inicio para los usuarios registrado. En este momento está autorizado para votar los cursos y marcarlos como interesantes para él.
---	--

Tabla 5 - caso de uso 4

- **Flujo alternativo:**

6 – El sistema detecta que el usuario se ha equivocado al introducir la contraseña, o el correo electrónico no consta en la base de datos. El sistema muestra un mensaje de advertencia y le solicita al usuario que vuelva a introducir las credenciales.

DSS: *Identificarse en el sistema*



powered by Astah

Figura 6 - caso de uso 4

#### 2.4.5 Caso de uso: Recuperar contraseña

Hoy en día la carga mental que existe sobre los usuarios y las contraseñas es abrumadora: el pin de la tarjeta de crédito, el pin del teléfono, las contraseñas de las 10 cuentas de correo, personales y de trabajo, el pin de la alarma, las claves de Facebook, Twitter, y así hasta el infinito. Es necesario disponer en cualquier sistema por pequeño que sea la posibilidad a sus usuarios de recuperar esa contraseña olvidada.

- **Actor principal:** Usuario registrado

- **Precondiciones:** El usuario está registrado, recuerda el correo electrónico con el que se registró, pero no recuerda la contraseña.
- **Postcondiciones:** El usuario cambia su contraseña y se identifica en el sistema.
- **Escenario principal de éxito:**

Acción del actor (Usuario registrado)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El usuario no recuerda su contraseña y pulsa sobre el enlace que tiene el literal ‘¿Olvidó su contraseña?’.	4 - El sistema le muestra una etiqueta input para que introduzca el correo electrónico con el que se registró en su momento y un botón para las instrucciones.
5 - El usuario rellena la casilla y pulsa el botón con el literal “Envíame las instrucciones para resetear la contraseña”.	6 - El sistema envía un correo electrónico con un enlace único para cambiar la contraseña, con el cual el sistema sabe a qué usuario pertenece y por un tiempo limitado. Y muestra la página de identificación.
7 - El usuario recibe el correo pulsa sobre el enlace, y este a su vez abre un navegador con una dirección especial. El usuario escribe su nueva contraseña por dos veces y pulsa el botón “cambiar mi contraseña”.	8 – El sistema cambia la contraseña del usuario registra su entrada y muestra la página de inicio del usuario registrado.

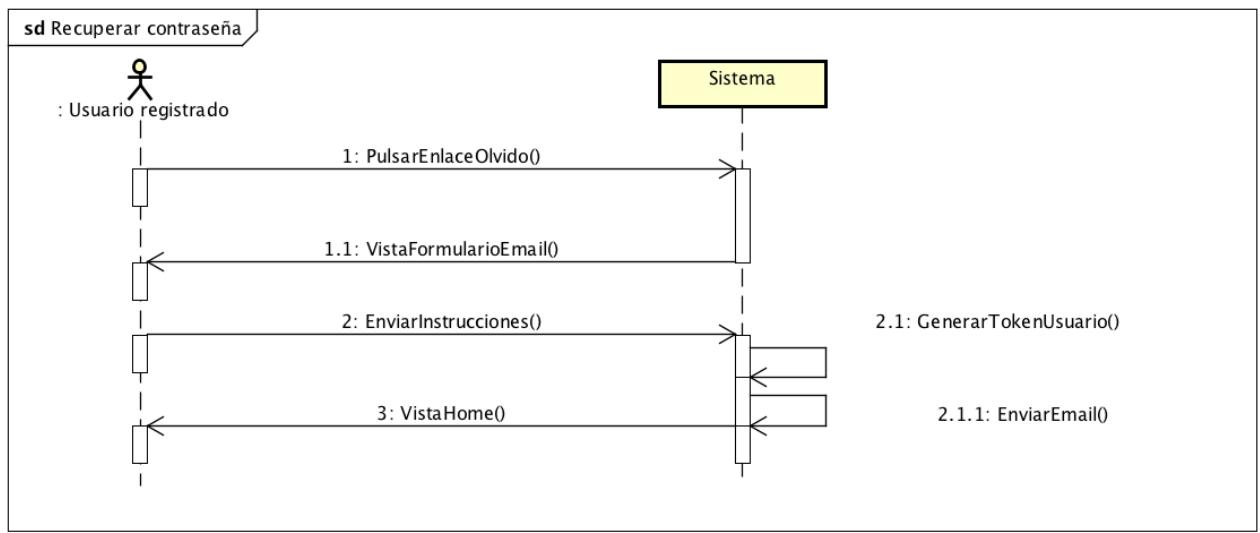
Tabla 6 - caso de uso 5

- **Flujo alternativo:**

6 - El sistema detecta que el correo electrónico no es válido, o no consta en nuestro sistema.

7 – Se muestra un mensaje pidiendo al usuario que introduzca un correo electrónico válido.

## DSS: Recuperar contraseña



powered by Astah

Figura 7 - caso de uso 5

### 2.4.6 Caso de uso: Cambiar contraseña

Todo usuario debe poder cambiar su contraseña, bien porque se ha visto comprometida tal contraseña, o porque le parece muy sencilla de descubrir, o porque se la ha proporcionado a alguien de su entorno y ahora está arrepentido. Es muy importante proporcionar un mecanismo para la sustitución de las contraseñas a petición del usuario.

- **Actor principal:** Usuario registrado
- **Precondiciones:** El usuario está identificado en el sistema y posicionado en la página de inicio para usuarios registrados.
- **Postcondiciones:** El sistema ha registrado el cambio de contraseña para el usuario que lo ha solicitado.
- **Escenario principal de éxito:**

Acción del actor (Usuario registrado)	Responsabilidad del sistema
1 - El usuario se identifica en el sistema según el caso de uso anteriormente citado.	2 - El sistema muestra la página de inicio para los usuarios registrados.
3 - El usuario pulsa sobre el enlace con su dirección de correos.	4 - El sistema le muestra un formulario para editar sus datos: email y contraseña. También muestra un botón para volver atrás y otro para cancelar la cuenta en el sistema.
5 - El usuario rellena la etiqueta con su nueva contraseña, confirma que no se equivoca	6 - El sistema comprueba que las dos contraseñas escritas coinciden, realiza el

introduciendo de nuevo la misma contraseña en otra etiqueta, también debe proporcionar la contraseña actual para confirmar que no es una usurpación de identidad, y pulsa el botón con el literal “Actualizar”.	cambio de la misma y muestra la página de inicio para usuarios registrados.
---	---

Tabla 7 - caso de uso 6

- **Flujo alternativo:**

6 - El sistema detecta que la contraseña introducida es más corta de lo exigido, seis caracteres como mínimo.

7 - El sistema muestra un mensaje e indica al usuario que vuelva a introducir una contraseña válida.

DSS: Cambiar contraseña

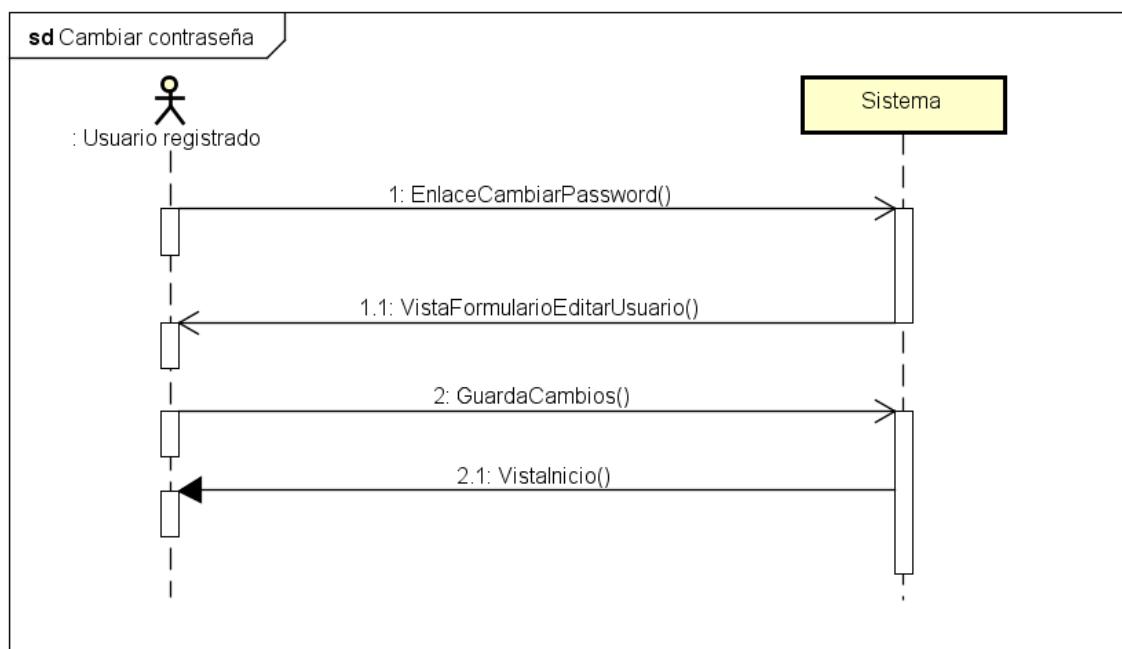


Figura 8 - caso de uso 6

#### 2.4.7 Caso de uso: Votar un curso

En este proyecto el uso de la experiencia proporcionado por otros usuarios de la comunidad es parte fundamental para la pormenorización de los cursos (ranking). Por ello se ha diseñado un sistema de evaluación de los cursos, que oriente a los próximos candidatos que quieran realizar dicho curso tendrán un prejuicio más ajustado a la realidad. Esta evaluación se ha implementado siguiendo los principios del Diseño Universal de Aprendizaje (UDL – Universal Learning Design).

- **Actor principal:** Usuario registrado
- **Precondiciones:** El usuario debe tener una cuenta activa en el sistema.
- **Postcondiciones:** El usuario se identifica en el sistema y evalúa un curso MOOC que haya realizado.
- **Escenario principal de éxito:**

Acción del actor (Usuario registrado)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El usuario hace clic en el enlace “Identificarse”.	4 - El sistema le muestra la página para identificarse en la aplicación, donde existe una etiqueta input para que introduzca el correo electrónico y otra para que introduzca la contraseña.
5 - El usuario rellena las casillas y pulsa el botón con el literal “Identificarse”.	6 - El sistema comprueba que es un usuario valido, registra su entrada y le muestra la página de inicio para usuarios registrados.
7 - El usuario el usuario hace una búsqueda del curso en el que está interesado, ver caso de uso “Buscar curso en el sistema”, para mayor detalle del mismo.	8 - El sistema muestra los resultados obtenidos.
9 - El usuario hace clic en el icono con el dedo pulgar hacia arriba.	10 - El sistema muestra un formulario con varias preguntas puntuables del 1 al 5, y algunas etiquetas de texto para aportar valoraciones.
11 - El usuario rellena el cuestionario y pulsa el botón con el literal “Crear evaluación”.	12 - El sistema registra su evaluación y muestra el curso evaluado con su nueva evaluación.

Tabla 8 - caso de uso 7

- **Flujo alternativo:**

11 – El usuario pulsa sobre el botón cancelar.

12 - El sistema vuelve a mostrar el detalle de curso que se estaba evaluando y descarta las puntuaciones que el usuario pudiera haber marcado para este curso.

### DSS: Votar un curso

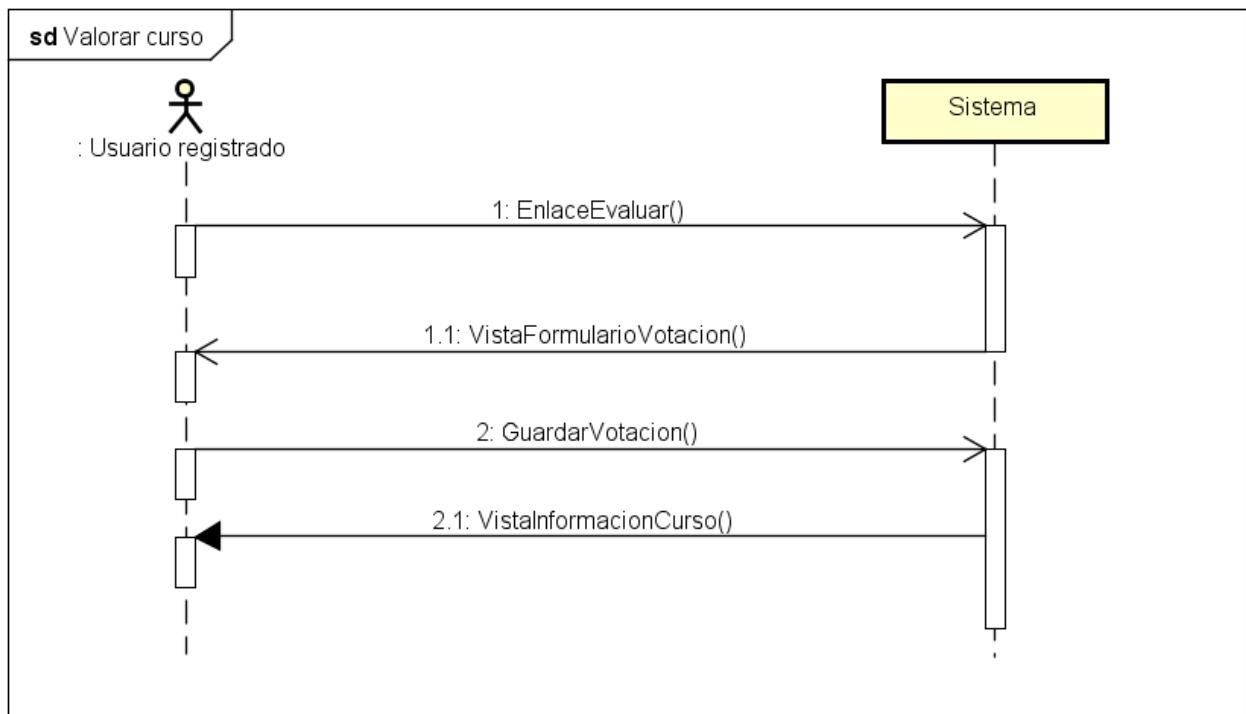


Figura 9 - caso de uso 7

#### 2.4.8 Caso de uso: Seleccionar un curso como interesante

Esta funcionalidad sirve para marcar un curso como de interés, aunque no se disponga en este momento de tiempo para realizarlo. También sirve para que en el hipotético caso de que el usuario se encuentre indeciso entre varios cursos, el sistema los guarda para una posterior decisión.

- **Actor principal:** Usuario registrado
- **Precondiciones:** El usuario debe tener una cuenta activa en el sistema y estar identificado en el mismo.
- **Postcondiciones:** El usuario marca un curso MOOC como interesante, para que el sistema se lo muestre al usuario cada vez que esté en su página de inicio.
- **Escenario principal de éxito:**

Acción del actor (Usuario registrado)	Responsabilidad del sistema
1 - El usuario entra en la página de inicio y se identifica y busca un curso de su interés, casos de uso “Identificarse en el sistema” y “Buscar curso en el sistema”.	2 - El sistema realiza las acciones detalladas en los casos de uso mencionados en el punto 1.

<p>3 - El usuario hace clic en el botón con un icono de un ojo abierto.</p>	<p>4 - El sistema añade el curso en la lista de interés para el usuario, y muestra la página de inicio para el usuario registrado, con la lista de cursos que ha marcado con anterioridad.</p>
---	--

Tabla 9 - caso de uso 8

- **Flujo alternativo:**

No se contempla ningún flujo alternativo.

DSS: *Seleccionar un curso como interesante*

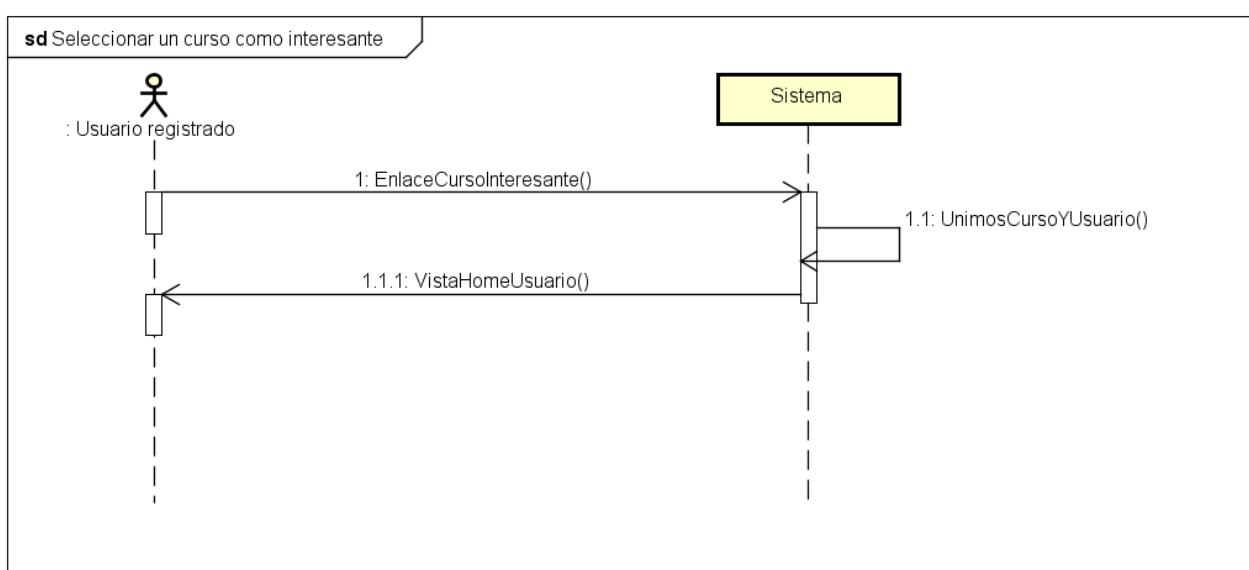


Figura 10 - caso de uso 8

#### 2.4.9 Caso de uso: Adquirir información en las diferentes plataformas

Este es uno de los pilares del sistema desarrollado, por ello se va a dividir su funcionamiento en dos casos de uso, uno será este que trata la relación entre el administrador que lanza la tarea y las respuestas del sistema. Más adelante, otro caso de uso lo tratará desde el punto de vista del procedimiento para obtener la información deseada.

- **Actor principal:** Administrador
- **Precondiciones:** El administrador debe estar dado de alta en el sistema como tal y tener un dispositivo con un navegador instalado y una conexión a internet.
- **Postcondiciones:** Las bases de datos del sistema son llenadas por toda la información adquirida y analizada por el sistema, como, por ejemplo: los cursos, las instituciones, las plataformas, etc.
- **Escenario principal de éxito:**

Acción del actor (Administrador)	Responsabilidad del sistema
1 - El administrador entra en la página de inicio (home).	2 - El sistema muestra la página de inicio para los usuarios no identificados.
3 - El administrador hace clic en el enlace "Identificarse".	4 - El sistema le muestra la página para identificarse en la aplicación, donde existe una etiqueta input para que introduzca el correo electrónico y otra para que introduzca la contraseña.
5 - El administrador rellena las casillas con las credenciales del administrador y pulsa el botón con el literal "Identificarse".	6 - El sistema comprueba que es un administrador valido, registra su entrada y le muestra la página de inicio para administradores registrados, donde se puede ver un gráfico de las puntuaciones realizadas y unos botones para la adquisición y análisis de curso MOOC en la web.
7 - El administrador el usuario hace clic en uno de los botones de adquisición y análisis.	8 – El sistema muestra un mensaje indicando que el procedimiento de adquisición se ha puesto en marcha y lanza una tarea en diferido para no interrumpir el trabajo del administrador.

Tabla 10 - caso de uso 9

- **Flujo alternativo:**

- 7 - El sistema identifica el rol de la persona que está accediendo al sistema como usuario, no como administrador
- 8 - El sistema muestra la página de inicio para usuarios registrados.
- 9 - El usuario no puede realizar la tarea de adquisición de datos, porque no está autorizado.

## DSS: Adquirir información en las diferentes plataformas

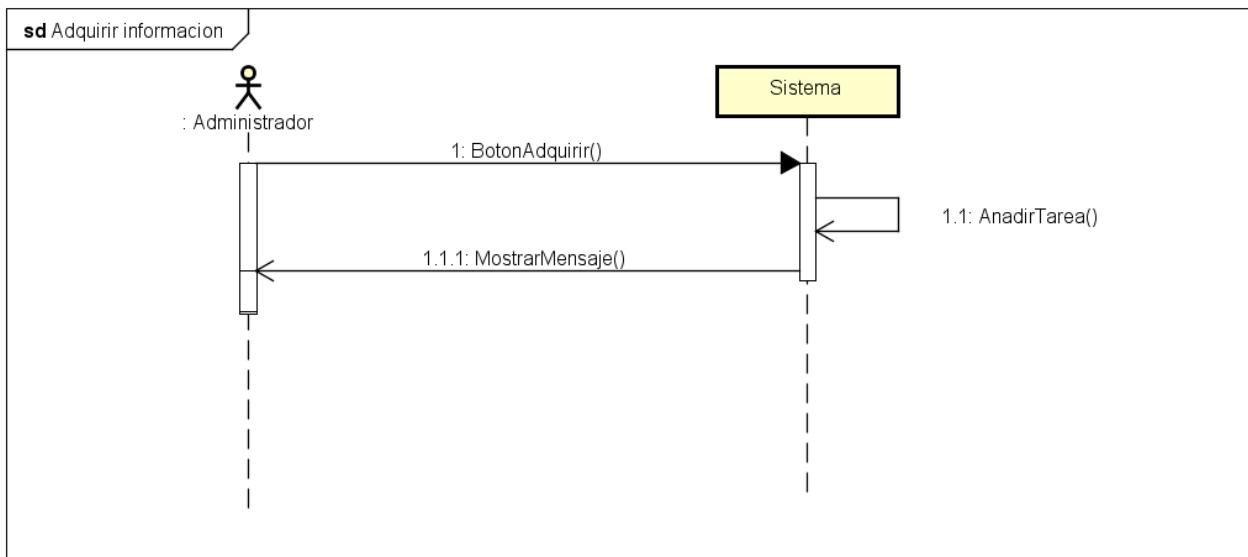


Figura 11 - caso de uso 9

### 2.4.10 Caso de uso: Gestionar los cursos

Todo el trabajo que el sistema no ha podido realizar de forma autónoma y automática, deberá ser supervisada y en caso de necesidad modificada por la persona que los gestiona (el administrador designado para ello). Entre estas tareas estará la modificación de datos extraídos de las diferentes plataformas, la relación entre cursos de diferentes ediciones, añadir subtítulos al curso, marcar como no disponible para las búsquedas, etc.

- **Actor principal:** Administrador
- **Precondiciones:** El administrador debe estar identificado en el sistema.
- **Postcondiciones:** Quedan reflejados los cambios solicitados por el administrador sobre un curso.
- **Escenario principal de éxito:**

Acción del actor (Administrador)	Responsabilidad del sistema
1 - El administrador entra en la página de inicio.	2 - El sistema muestra la página de inicio para los administradores.
3 - El administrador hace clic en el enlace de la barra superior con el literal "Cursos".	4 - El sistema le muestra al administrador la página con todos los cursos adquiridos que tiene el sistema, paginados en bloques de 20 cursos por página. El administrador puede seleccionar una página en concreto, un enlace al detalle de un curso, realizar una búsqueda de

	un curso en concreto o hacer clic sobre el botón para la creación de cursos de forma manual.
5 - El administrador hace clic sobre el enlace de un curso.	6 - El sistema muestra los detalles de la información de la que dispone sobre el curso, junto con unos botones para la edición de los datos del curso, el borrado del curso, el añadido de una nueva transcripción, audio y subtítulo para el curso. También muestra los comentarios añadidos para este curso de los usuarios que lo han evaluado.
7 - El administrador hace clic en el botón de edición.	8 – El sistema muestra un formulario con los datos del curso para su modificación, y unos botones en la parte inferior para actualizar los cambios o cancelarlos.
9 - El administrador realiza los cambios que estime oportunos y hace clic sobre el botón “Actualizar curso”.	10 - El sistema guarda los cambios proporcionados por el administrador y muestra la página de inicio del administrador.
11 - El administrador pulsa el botón de la esquina superior derecha para salir del sistema.	12 - El sistema registra su salida y muestra la página inicial para los usuarios sin registrar.

Tabla 11 - caso de uso 10

- **Flujo alternativo:**

9 – El administrador decide cancelar los cambios realizados, pulsando sobre el botón “Cancelar”.

10 - El sistema vuelve a la página de información del curso que ha sido editado.

#### DSS: Gestionar los cursos

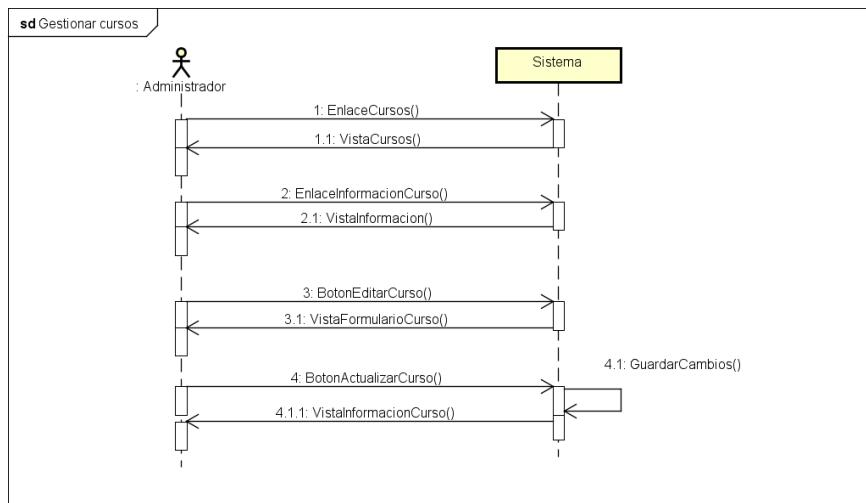


Figura 12 - caso de uso 10

#### 2.4.11 Caso de uso: Gestionar las Instituciones, Plataformas e Idiomas

En el escenario principal se hace referencia a las instituciones, pero este escenario principal es compartido en el caso de las plataformas e idiomas, por lo que se han unificado como un único caso de uso. Las instituciones y las plataformas son dadas de alta de forma automática por el sistema cuando realiza las técnicas de adquisición y análisis de las plataformas, o también conocido como Harvesting.

- **Actor principal:** Administrador
- **Precondiciones:** El administrador debe estar identificado en el sistema y ubicado en la página de inicio para administradores.
- **Postcondiciones:** Quedan reflejados los cambios solicitados por el administrador sobre un curso.
- **Escenario principal de éxito:**

Acción del actor (Administrador)	Responsabilidad del sistema
1 - El administrador hace clic en el triángulo invertido de la barra superior, entre el botón de los Cursos y la etiqueta de búsqueda.	2 - El sistema despliega un menú en el que contiene enlaces para la gestión de las Instituciones, Plataformas e Idiomas.
3 - El administrador hace clic sobre el enlace de las Instituciones. En el caso de las Plataformas e Idiomas el procedimiento es el mismo.	4 - El sistema muestra una página con todos las Instituciones paginadas en bloques de 20, los nombres de las instituciones son a su vez enlaces para la edición de la institución y un botón para añadir nuevas instituciones.
5 - El administrador hace clic sobre el nombre de una institución a la que desea hacer algún cambio.	6 - El sistema muestra el nombre de la institución, los botones de "Atrás", "Editar" y "Borrar". También detalla información de los cursos donde está relacionada esta institución.
7 - El administrador hace clic en el botón de borrado.	8 - El sistema muestra un cambio de literal "¿Estás seguro?", para la confirmación del borrado.
9 - El administrador hace clic sobre el botón de borrado otra vez.	10 - El sistema borra la institución y muestra la página con la relación de instituciones anteriormente mencionada.

Tabla 12 - caso de uso 11

- **Flujo alternativo:**

9 - El administrador realiza una espera superior a 10 segundos.

10 - El sistema vuelve a mostrar el mensaje “Borrar” en el botón de borrado y cancela la petición anterior de borrado.

#### DSS: Gestionar las Instituciones, Plataformas e Idiomas

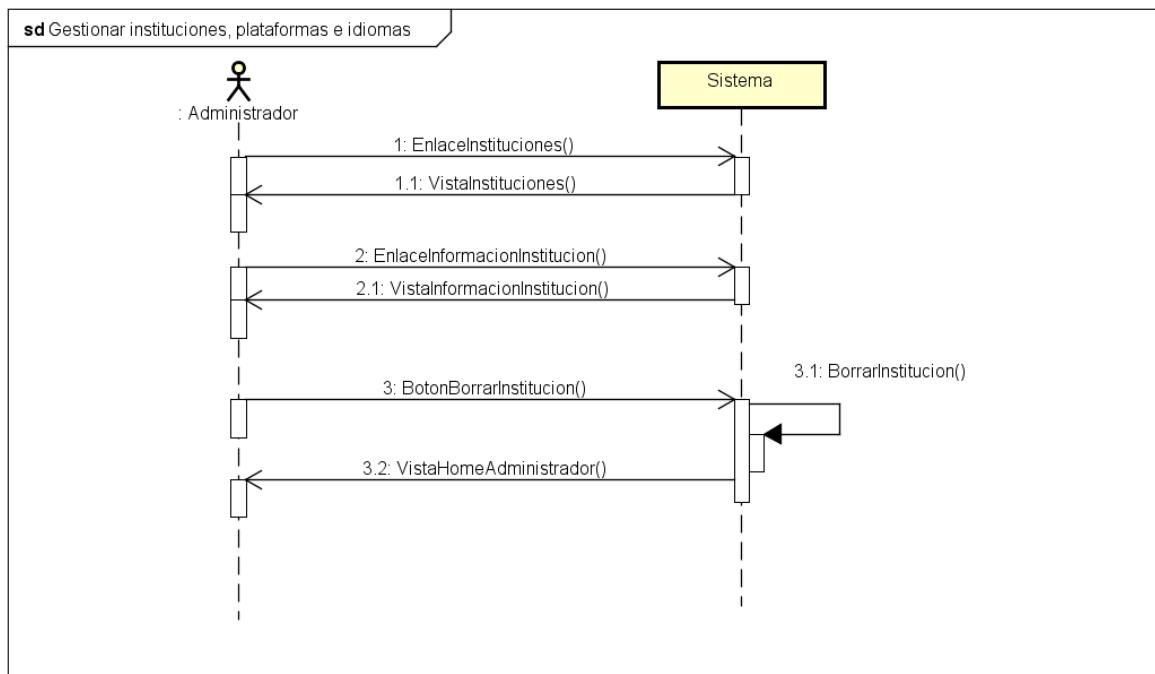


Figura 13 - caso de uso 11

#### 2.4.12 Caso de uso: Gestionar edición anterior

En este caso de uso es donde el administrador relaciona una anterior edición de un curso. Las puntuaciones obtenidas por cursos anteriores relacionados con este, lo que se conoce como ediciones anteriores de un curso, debe ser compartida por todas las ediciones posteriores.

- **Actor principal:** Administrador
- **Precondiciones:** El administrador debe estar identificado en el sistema, deben existir dos cursos o más que están relacionados entre sí, es decir, que sean diferentes ediciones de un mismo curso, y situado en la página de información detallada del curso con el número de edición más alto.
- **Postcondiciones:** En el sistema queda constancia de esa relación que hay entre los cursos de diferentes ediciones.
- **Escenario principal de éxito:**

Acción del actor (Administrador)	Responsabilidad del sistema
1 - El administrador entra en la página de edición de un curso haciendo clic sobre el botón “Editar”.	2 - El sistema muestra la página con el formulario de edición para el curso.

3 - El administrador hace clic sobre el desplegable situado al final del formulario, con la etiqueta “Edición anterior”.	4 - El sistema despliega las coincidencias más cercanas al curso que se está editando, con esto se evita que el desplegable contenga todos los cursos de los que se dispone en el sistema.
5 - El administrador hace clic sobre el curso que es la edición anterior de este mismo curso y pulsa sobre el botón actualizar curso.	6 - El sistema refleja esa relación entre los cursos con una lista enlazada.

Tabla 13 - caso de uso 12

- **Flujo alternativo:**

4 - El sistema no puede llenar el desplegable porque no existe ninguna coincidencia cercana al literal del título de curso.

5 - El administrador es incapaz de seleccionar nada puesto que el desplegable está vacío, y pulsa el botón “Cancelar”.

6 – El sistema muestra la página con la información detallada del curso.

DSS: *Gestionar edición anterior*

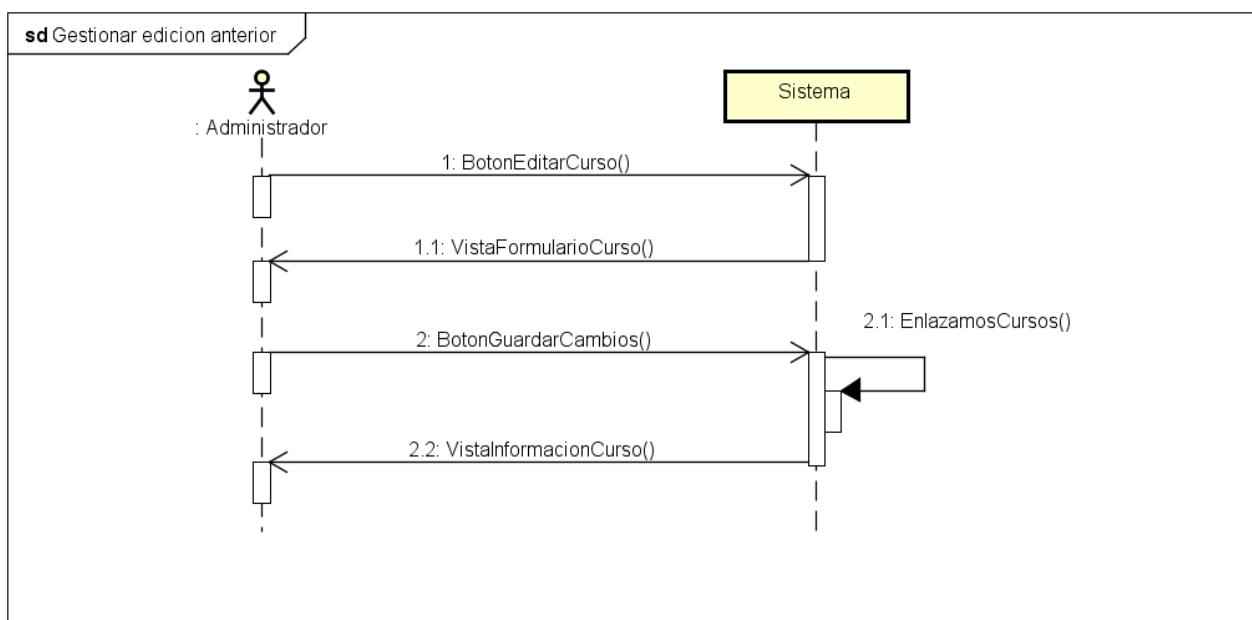


Figura 14 - caso de uso 12

#### 2.4.13 Caso de uso: Gestionar los usuarios

Los usuarios se darán de alta de forma autónoma, pero habrá ocasiones en que se desee dar de alta a un usuario, por ejemplo, por parte del administrador. Por ello se han incluido estas acciones tales como: añadir usuarios, borrar usuarios, cambiar el rol del usuario, modificar sus datos u obtener información de las valoraciones realizadas.

- **Actor principal:** Administrador
- **Precondiciones:** El administrador debe estar identificado en el sistema.
- **Postcondiciones:** En el sistema se refleja el cambio de rol para un usuario.
- **Escenario principal de éxito:**

Acción del actor (Administrador)	Responsabilidad del sistema
1 - El administrador hace clic en el triángulo invertido de la barra superior, entre el botón de los Cursos y la etiqueta de búsqueda.	2 - El sistema despliega un menú en el que contiene enlaces para la gestión de las Instituciones, Plataformas e Idiomas.
3 - El administrador hace clic sobre el enlace de los usuarios.	4 - El sistema muestra una página con todos los usuarios paginados en bloques de 20, también se muestran los perfiles que tiene cada usuario asignado.
5 - El administrador hace clic sobre el nombre de un usuario al que desea hacer algún cambio, en este caso, un cambio de rol.	6 - El sistema muestra el nombre del usuario, los botones de "Atrás", "Editar" y "Borrar". También detalla información de los cursos que ha evaluado este usuario.
7 - El administrador hace clic en el botón "Editar".	8 - El sistema muestra un formulario para sus datos y unas listas de acciones (check-box) para su rol.
9 - El administrador hace clic sobre los roles deshabilitando uno, habilitando otro y pulsando sobre el botón "Actualizar usuario".	10 - El sistema actualiza la información sobre este usuario en las bases de datos del sistema.

Tabla 14 - caso de uso 13

- **Flujo alternativo:**

- 9 - El administrador hace clic sobre el botón "Cancelar".
- 10 - El sistema descarta los cambios realizados en el usuario y vuelve a mostrar la página de los usuarios, paso 4.

## DSS: Gestionar los usuarios

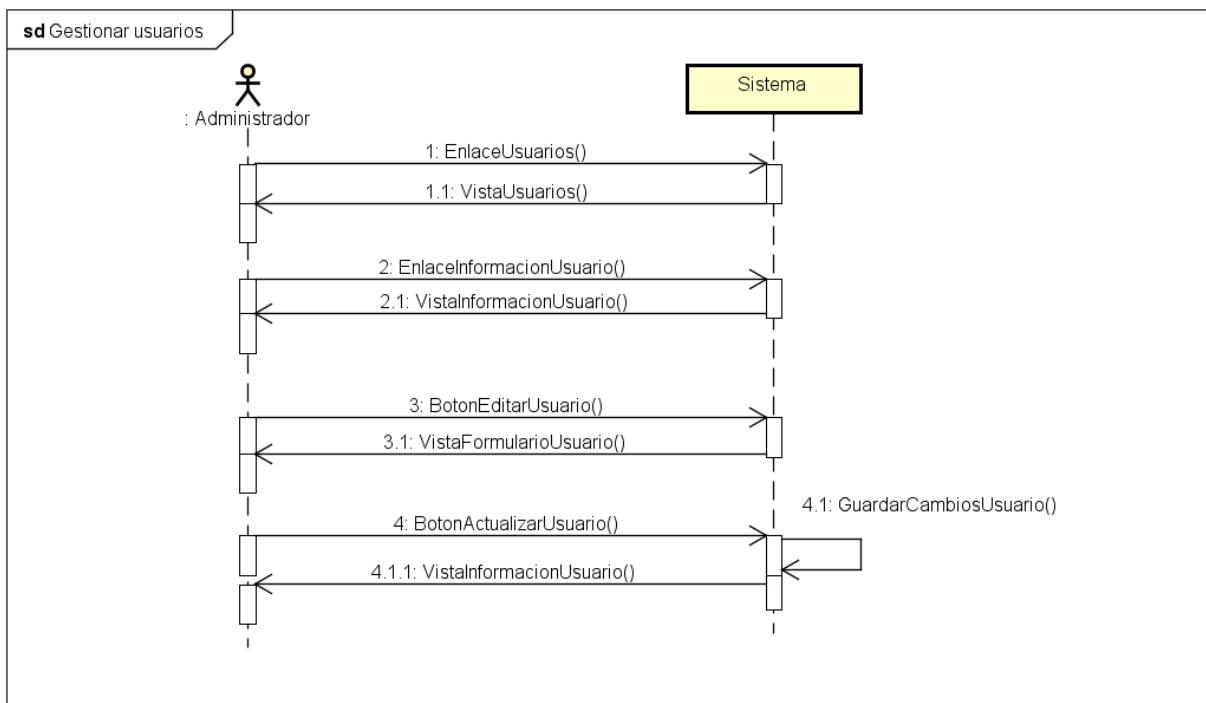


Figura 15 - caso de uso 13

### 2.4.14 Caso de uso: Adquirir y analizar cursos MOOC

En un caso de uso anterior ya se ha realizado una aproximación desde el punto de vista del administrador, ahora se hace desde el punto de vista del procedimiento llamado. Este es uno de los casos de uso más importantes del sistema, trata de recolectar la mayor y más fiable información sobre los cursos ofrecidos por las diferentes plataformas e impartidos por las instituciones. Trata de analizar de la mejor forma posible la información extraída de estos sitios web, para quitarle la mayor carga posible a la persona asignada para supervisar esta información, que en este caso será un administrador del sitio.

La tarea será lanzada de forma automática una vez al día o cuando el administrador estime oportuno. Los cursos obtenidos nunca deben ser repetidos, si ya se encuentran registrados en el sistema, este algoritmo debe desestimarlo.

- **Actor principal:** Sistema
- **Precondiciones:** El sistema recibe una petición del administrador para que realice la adquisición de información en las diferentes plataformas configuradas.
- **Postcondiciones:** Las bases de datos del sistema son llenadas por toda la información adquirida y analizada por el sistema, como, por ejemplo: los cursos, las instituciones y las plataformas.
- **Escenario principal de éxito:**

Acción del actor (Tarea diaria)	Responsabilidad del sistema
1 – Se lanza una tarea diaria o bien el sistema tiene registrada una petición del administrador para que realice la adquisición ahora mismo.	2 - El sistema lanza el algoritmo para la adquisición de los datos, Harvesting, en una plataforma.
	3 – El sistema identifica la lista de cursos ofertados por la plataforma y comienza un bucle para extraer la información uno por uno.
	4 - Este algoritmo lee las páginas web donde tienen alojados los cursos las diferentes plataformas, obteniendo el HTML completo de estos sitios.
	5 - Nuestro algoritmo analiza las etiquetas y guarda la información contenida en ellas en la base de datos, relacionando cada etiqueta HTML con un campo en concreto de nuestro sistema.
	6 - Si fuera necesario realizará un cambio de página como si de un usuario se tratara.
	7 – Una vez terminado el listado de cursos de la plataforma, la ejecución del algoritmo termina.

Tabla 15 - caso de uso 14

- **Flujo alternativo:**

7 – Una vez terminado el listado de cursos de la plataforma actual, se continúa con la siguiente plataforma de la lista de plataformas, en el paso 1, así hasta terminar con todo el listado de plataformas analizadas.

DSS: Adquirir y analizar cursos MOOC

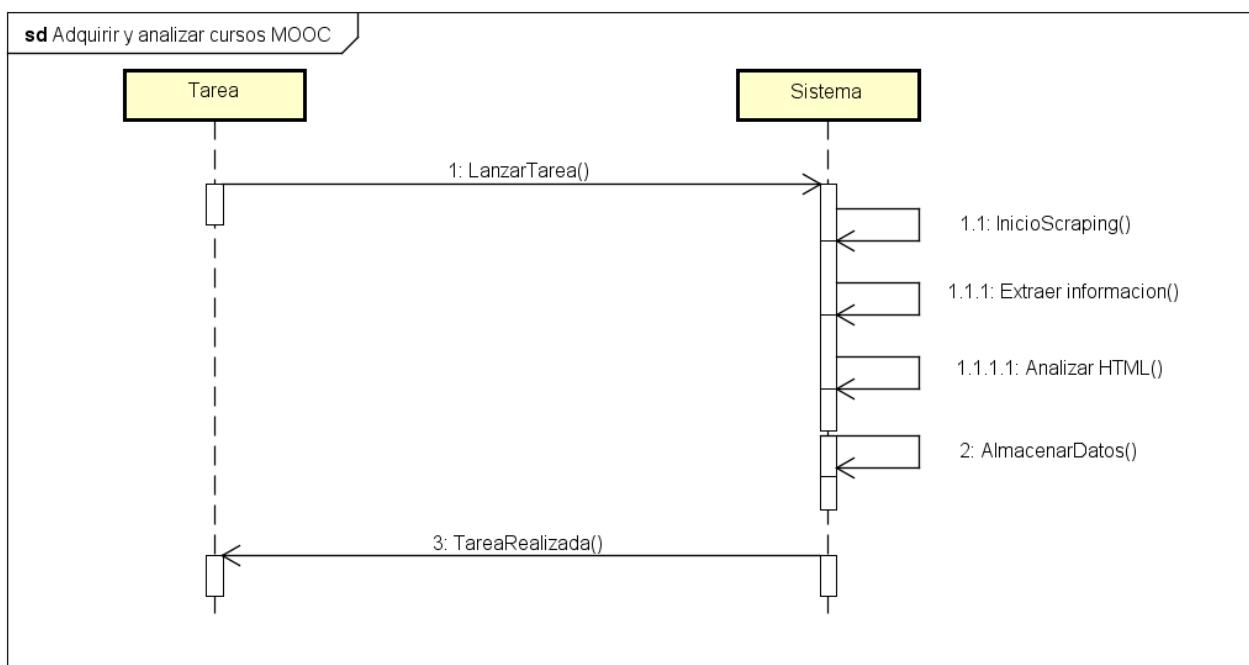


Figura 16 - caso de uso 14

## 2.5 Modelo conceptual

Con el siguiente modelo conceptual se puede apreciar la relación que va a existir entre las diferentes clases, también muestra una pincelada de cómo será, más o menos, la vista que representará dicha clase.

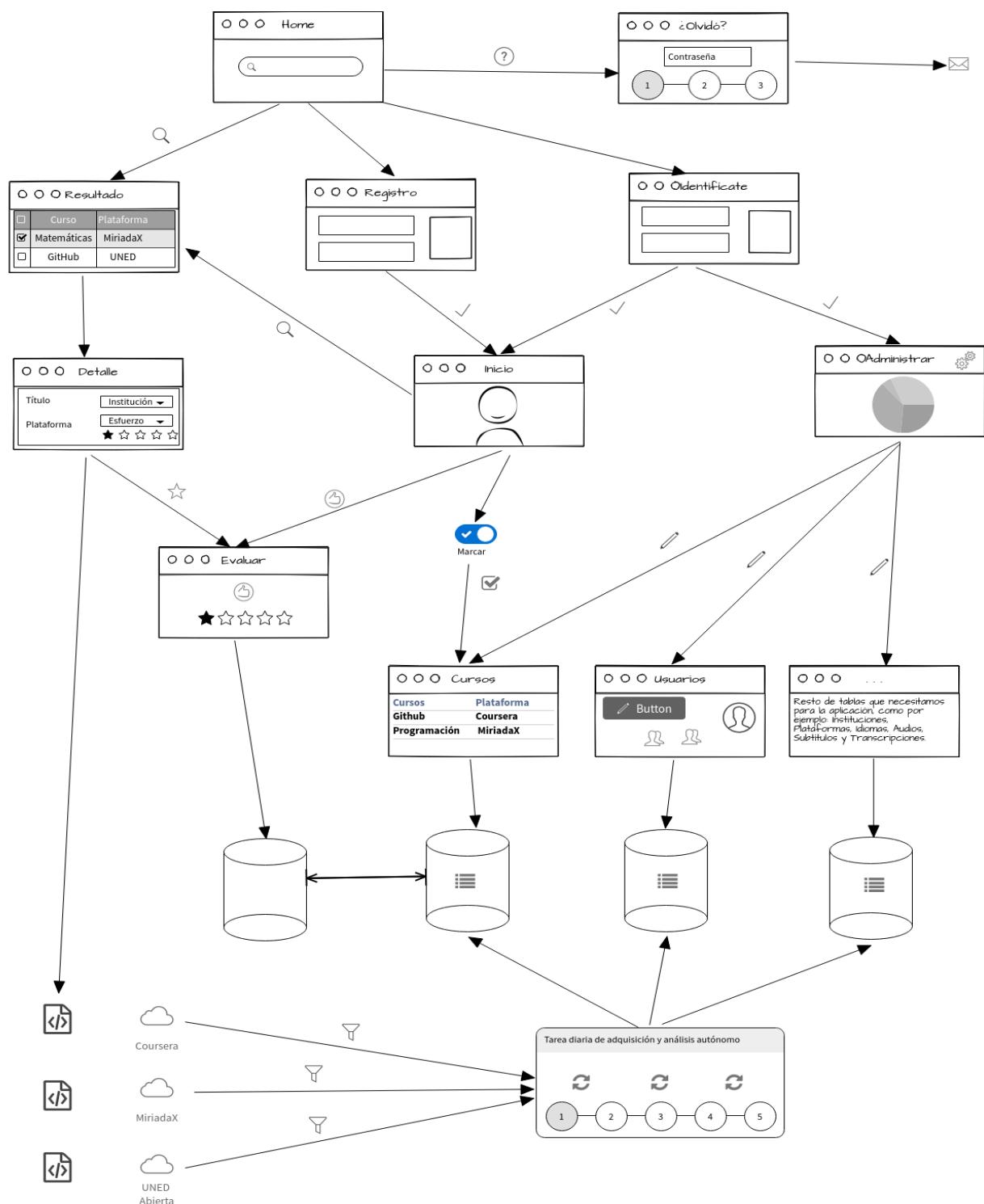


Figura 17 - Modelo conceptual 1

A continuación, se muestra otra visión del modelo conceptual.

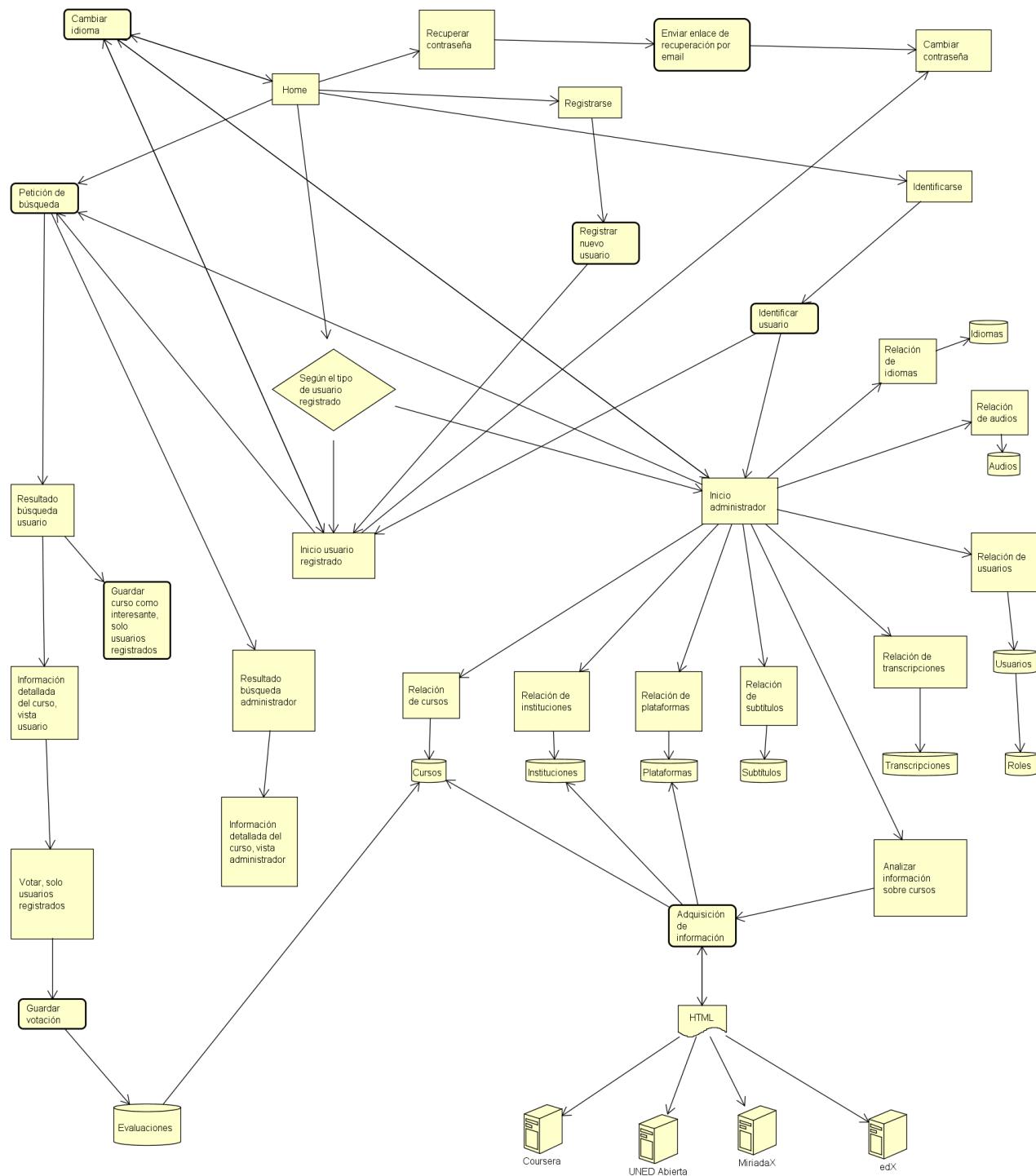


Figura 18 - Modelo conceptual 2

## **2.6 Requisitos funcionales**

1. Cualquier usuario del sistema puede hacer búsquedas sobre los cursos registrados en la base de datos.
2. El administrador puede añadir, borrar o modificar cursos de manera manual y aportar datos que el sistema no haya adquirido de forma autónoma.
3. El administrador también tendrá la capacidad de administrar las cuentas de los usuarios como, por ejemplo: crear, borrar y modificar datos de usuario.
4. El administrador podrá marcar cursos para que no puedan ser encontrados por el usuario, tanto registrado como anónimo.
5. Si el resultado de la búsqueda es superior a cuatro coincidencias, el sistema paginará los resultados en grupos de cuatro elementos.
6. El sistema debe contemplar múltiples idiomas, y el usuario lo podrá cambiar a conveniencia.
7. Cualquier usuario que lo solicite podrá ver la información detallada sobre un curso.
8. El usuario podrá registrarse en el sistema de una manera sencilla y automática, solo aportando el correo electrónico y la contraseña.
9. Una vez registrado en el sistema, el usuario podrá cambiar su contraseña cuando lo desee.
10. En caso de olvido de la contraseña, el sistema debe enviar un correo electrónico al usuario con un enlace para poder cambiar la contraseña.
11. Si el usuario ya está registrado, podrá identificarse para tener el rol de usuario registrado.
12. Los usuarios registrados pueden valorar los cursos.
13. Si un usuario no registrado intenta valorar un curso se deben pedir sus credenciales.
14. Los usuarios registrados, en su página de inicio personalizada, tendrán una muestra de los cursos que ha marcado como interesantes para él.
15. Ningún usuario, registrado o no, tendrá acceso a la parte de la administración. Solo los administradores podrán acceder a esta parte del sistema.
16. El administrador deberá contar con los mecanismos adecuados para crear, borrar o modificar cualquier tabla con las que cuenta el sistema, como: Usuarios, Cursos, Plataformas, Instituciones, Audios, Idiomas, Subtítulos y Transcripciones.
17. Existirá una tarea diaria en la que haya que adquirir, analizar y guardar la información que se extraiga de las plataformas oferentes de cursos MOOC.
18. Las tablas con información sobre los cursos serán llenadas de forma automática por el algoritmo de adquisición, como: Cursos, Plataformas, Instituciones e Idiomas
19. Se compartirán las puntuaciones de los usuarios han realizado sobre un mismo curso con diferentes ediciones.

## **2.7 Requisitos de seguridad**

Las siguientes funcionalidades se encuentran dirigidas a gestionar los aspectos relacionados con la seguridad que habrán de ser completados por el sistema.

1. Iniciar sesión, permitirá a los usuarios registrados identificarse en el sistema, para acceder a las funcionalidades ofrecidas en función de su tipología. La identificación en el sistema se realizará mediante la aportación de un correo electrónico y una contraseña.
2. Finalizar sesión, esta funcionalidad permite al usuario terminar la sesión actual, posibilitando la entrada de un nuevo usuario.
3. Cambio de clave, permitirá a un usuario, previamente identificado en el sistema, modificar su propia contraseña de acceso.
4. Alta de usuarios, permitirá a los administradores crear nuevos usuarios en el sistema, permitiendo, además, especificar su tipología (administrador o usuario).
5. Registro de usuarios, permitirá crear nuevos usuarios en el sistema con la tipología de usuario de forma autónoma. Similar al alta, pero sin intervención de un administrador.
6. Baja de usuarios, deberá posibilitar la eliminación de un usuario del sistema por parte de los administradores.
7. Modificación de usuarios, esta funcionalidad permitirá a los administradores modificar los datos de los usuarios registrados en el sistema.
8. Consulta de usuarios, deberá permitir a los administradores consultar información de los usuarios registrados en el sistema, como, por ejemplo: la última conexión realizada, evaluaciones realizadas, etc.

## 2.8 Requisitos no funcionales

El sistema deberá contar con un interfaz de usuario que se adapte al mayor número posible de dispositivos electrónicos, por ejemplo: ordenadores personales, teléfonos móviles, tabletas, etc.

1. No se requiere ningún sistema operativo específico, pero sí que ese sistema operativo cuente con un navegador instalado y conectado a Internet. Tampoco es necesario que tenga una versión actualizada del navegador porque al ser HTML, CSS y JavaScript, se puede ejecutar en cualquier navegador independientemente de la versión del mismo.
2. Las interfaces de usuario deben ser operativas aún sin CSS activo, esto es así porque el sitio web debe ser accesible con dispositivos como los lectores de pantalla u otros similares.
3. Al ser una aplicación web debe ser intuitiva, ya que el usuario que llega por primera vez al sitio web no dispone de manual de aplicación, ni tampoco de una persona que pueda indicarle el modo de usar la aplicación web.
4. Respecto al hardware necesario lo más importante a destacar sería el ancho de banda, ya que al ser una página web se corre el riesgo, de tener una concurrencia elevada, siendo utilizada por un gran número de personas al mismo tiempo. Si además se desea contar con alta disponibilidad habrá que contar con servicios alternativos de forma inmediata.
5. El sistema debe ser fiable, ya que al ser una aplicación web va a soportar un acceso concurrente muy elevado y no se puede permitir una caída del servidor, esto provocaría que los usuarios dejaran de usar la herramienta. La disponibilidad de la aplicación debe ser

plena y la seguridad es un requisito importante ya que contiene información de los usuarios.

6. El mantenimiento de la aplicación es un aspecto clave a largo plazo, ya que el desarrollo web sufre constantes avances y se tiene que adaptar el proyecto a estos avances en la red de forma continuada en el tiempo.

# 3. Diseño

En este apartado se tratará el diseño realizado para este proyecto, cumpliendo con los requisitos detallados en el apartado anterior. El diseño realizado se divide en seis apartados, que van desde la arquitectura utilizada hasta el sistema de almacenamiento permanente de los datos utilizados en el sistema.

## 3.1 Arquitectura del sistema

Para este proyecto se han utilizado varias arquitecturas, como, la arquitectura Cliente-Servidor, el patrón de arquitectura MVC, Modelo Vista Controlador y el modelo REST de Transferencia de Estado Representacional.

### 3.1.1 Modelo Cliente-Servidor

En este modelo de arquitectura hay dos partes claramente diferenciadas, por un lado, el cliente, que es la parte del modelo que realiza peticiones, y el servidor es el encargado de responder a esas peticiones.

El cliente gestiona las funcionalidades relacionadas con la manipulación y representación de la información gestionada, también gestiona toda la interacción con el usuario del modelo. Por eso están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario lo más productivas posibles, conocido como front-end.

El servidor es el encargado de atender todas las peticiones realizadas por los clientes a los recursos gestionados por él. Esta parte de la arquitectura se la conoce como back-end. También es la parte de la arquitectura encargada de gestionar todas las funcionalidades relacionadas con el modelo de negocio y la responsable del almacenamiento de los datos generados.

En resumen, la arquitectura Cliente-Servidor es una relación entre procesos corriendo, casi siempre, en máquinas separadas. El servidor es un proveedor de servicios y el cliente es un consumidor de servicios. Interactuando por un mecanismo de intercambio de mensajes. En nuestro sistema las peticiones realizadas desde el cliente al servidor y las respuestas desde el servidor al cliente serán mediante http.

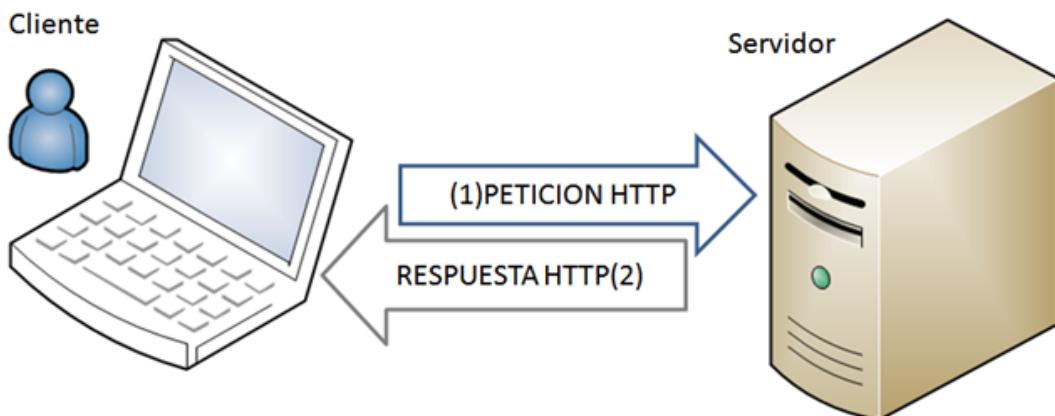


Figura 19 - Cliente-servidor<sup>13r</sup>

### 3.1.2 Modelo Vista Controlador (MVC)

Este modelo consiste en separar nuestro desarrollo en tres partes bien diferenciadas: los datos de la aplicación, la interfaz de usuario y la lógica de negocio. Estos tres componentes diferenciados se relacionarán para que al final se obtiene una aplicación robusta.

En la parte del Modelo se hace referencia a todo lo relacionado con las bases de datos: conexiones, extracción e introducción de información, campos calculados, relaciones entre diferentes tablas, etc. En el sistema esta parte estará controlada por ActiveRecord, las clases heredan todas las funcionalidades de este ORM. Aquí se deposita todo el código relacionado con el manejo de información a nivel de bases de datos.

La Vista contiene todo el código necesario para generar y controlar la interfaz que va a interactuar con el usuario. En este caso se utiliza HTML para la maquetación, CSS para darle forma a la interfaz y JavaScript para manipular del DOM, en la parte del navegador Web.

El Controlador es el centro de control para todo el modelo, recibe las peticiones del usuario a través de los eventos, realiza las solicitudes a la parte del Modelo (ActiveRecord) manipula la información obtenida del modelo de negocio, rellena de contenido las Vistas y por último envía al usuario la Vista obtenida, en el caso una página HTML.

---

<sup>13</sup> Pedro Herrera Sánchez, Fundamentos de funcionamiento de una aplicación web.  
Recuperado de <http://www.devjoker.com/contenidos/catss/518/mapaweb.aspx>

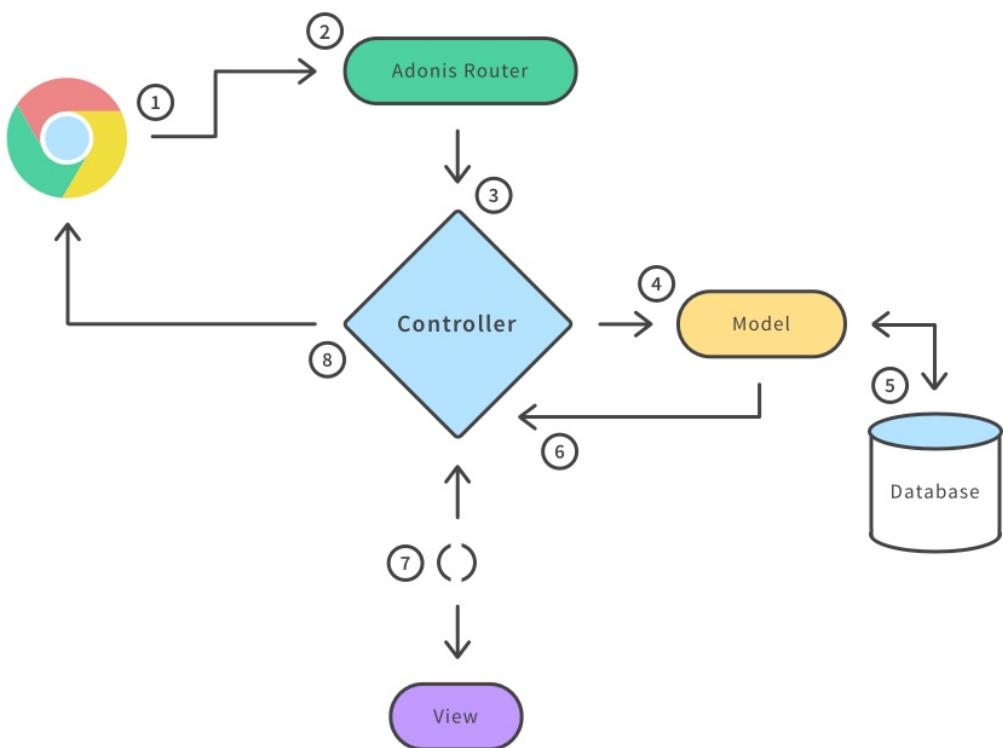


Figura 20 - modelo/vista/controlador<sup>14</sup>

Por tanto, estos serían los pasos que seguir para poder enviar peticiones al servidor y obtener respuestas del mismo.

1. El usuario desde un navegador realiza una petición al servicio Web: una búsqueda, una petición de información, un marcado de un curso, una evaluación, etc.
2. El navegador crea una solicitud http y la envía a la URL donde está alojado el servidor. Ya sea una solicitud para obtener información mediante el método GET o una solicitud de modificación mediante el método POST.
3. La petición http llega al Servicio Web, el Controlador, que la gestiona y si es necesario hace algún tipo de consulta para actualizar o recuperar información desde o hacia la base de datos, el Modelo.
4. El servidor crea una respuesta http que incluye la Vista generada desde el Servicio Web, enviándolo de vuelta a la dirección desde la que se recibió la petición.
5. El navegador recibe la respuesta del servidor y procesa la información obtenida.
6. El navegador muestra la Vista obtenida.

<sup>14</sup> ADONIS, Abhimanyu Rana y Harminder Virk, MVC Pattern.  
Recuperado de <https://www.adonisjs.com/docs/3.2/mvc-pattern>

### 3.1.3 Servicios REST

REST utiliza los métodos de http para realizar los requerimientos de información y retornar como cabeceras de la respuesta http en conjunto con la información requerida.

A continuación, los cuatro métodos de petición que existen en http:

1. El método GET, permite obtener información del servidor, especificando un ID para mostrar algún registro en particular o sin el ID para obtener una colección de varios registros.
2. El método POST, permite enviar información al servidor para ser almacenada, por ejemplo, un formulario con datos sobre un curso.
3. El método PUT, con este método se puede modificar un registro que se haya creado con anterioridad, por ejemplo, cuando se quiere editar un curso a través de un formulario y se indica al servidor que modifique un registro ya existente.
4. El método DELETE, no permite enviar un parámetro a través de la URL para borrar un registro en el servidor.

Utilizando estos cuatro métodos existe la posibilidad de indicar explícitamente al servidor la acción que se quiere ejecutar. REST también proporciona consistencia a través de algunas restricciones: la más importante es que no tiene estado, es decir, cada solicitud al servidor debe contener toda la información necesaria para que el servidor desde su contexto entienda de qué se trata la solicitud.

## 3.2 Guía de estilo y diseño

En las siguientes líneas se define la guía estilo y diseño utilizada en la aplicación web YourMOOC4all, también se muestra todos los puntos descritos en la página web, en referencia a la accesibilidad<sup>15 16 17</sup>.

La Estructura esta es una parte muy importante del sitio, si no se plantea correctamente el usuario se puede perder por el sitio, pueden quedar páginas por encontrar o simplemente el usuario abandonara el sitio por saturación de información. La estructura depende de manera directa con el contenido del sitio y hay que buscar la manera más fácil y clara de mostrar los contenidos del mismo. Puede ser: jerárquica, lineal, lineal con jerarquía o red. La estructura dependerá del tipo de usuario: si es un usuario, registrado o no, o si es un administrador. En caso de ser un usuario la página web mostrada siempre será parecida, cambiando únicamente el resultado obtenido en la búsqueda o si se está administrando el perfil del usuario. Por el

---

<sup>15</sup> [https://www.usableyaccesible.com/recurso\\_misvalidadores.php](https://www.usableyaccesible.com/recurso_misvalidadores.php)

<sup>16</sup> <http://accesibilidadweb.dlsi.ua.es/?menu=hr-trabajar-color>

<sup>17</sup> <https://addons.mozilla.org/es/firefox/addon/wcag-contrast-checker/>

contrario, si es un administrador el que está usando la plataforma, está contendrá un menú para la navegación dentro de la misma página. Este menú de navegación dentro de la propia página no hace falta duplicarlo porque es flotante y siempre está visible.

### 3.2.1 El esquema de página.

En relación con el tamaño se ha intentado que la página no ocupe más del doble de una pantalla estándar. El problema es que hay infinidad de resoluciones de pantalla y esto dificulta mucho calcular un alto correcto. También depende mucho de la cantidad de cursos que resulten de la búsqueda. En cada documento generado en la plataforma, se ha incluido el contacto del desarrollador y la fecha de creación de la página.

Los únicos enlaces al exterior son los enlaces a las plataformas que administran los cursos, los cuales se abren en una pestaña nueva del navegador y conducen al sitio donde se puede formalizar la inscripción en el curso que se está interesado. También el enlace a la dirección de correo electrónico hace una llamada nuestro gestor de correo electrónico con el correo electrónico del desarrollador en el apartado de destinatario. El funcionamiento de estos enlaces parece ser lo suficientemente llamativa. Se cumple con la independencia de navegador e incluso del dispositivo en el que la aplicación va a ser ejecutada. Está comprobado en Opera, Firefox, Chrome, Edge e Internet Explorer. Esto es debido al uso de técnicas de diseño adaptativas y del Framework Bootstrap.

En el apartado de tipografías siguiendo las recomendaciones del W3C se ha utilizado las etiquetas de encabezados h1, h2, h3 y los strong, para cambiar el tamaño y las negritas respectivamente. Al utilizar fuentes estándar se asegura que estén instaladas en todos los dispositivos utilizados para las pruebas. En caso de tener que optar por alguna tipografía en concreto sería Serif o Sans Serif ya que es la mejor tipografía para mostrar en las pantallas de los dispositivos. El apartado de gráficos está cubierto con las imágenes aportadas por el análisis de las diferentes plataformas y los cursos que allí se imparten. Todas cuentan con un ALT, con el nombre de curso al que pertenece la imagen. No se han utilizado imágenes en el fondo de la pantalla ni ninguna otra filigrana gráfica para que sea lo más accesible a los lectores de pantalla que usan las personas con deficiencia visual.

Para la aceleración y velocidad de carga de las diferentes páginas parece muy optimizada. Solo hay una ocasión en la que la carga de la aplicación web es sustancial, cuando lleva mucho tiempo sin haber ninguna petición a la aplicación, el servidor de Heroku pone la aplicación en modo de reposo, y cuando vuelve a entrar otra petición ésta tarda unos segundos en despertar. El formato utilizado que se ha utilizado para las imágenes del sitio son JPG, no es tan bueno como el PNG, pero es un estándar. No se han utilizado ningún tipo de banners para tener la mayor aceptación posible de los estándares de accesibilidad.

### 3.2.2 Accesibilidad.

En principio las diferentes páginas generadas por la aplicación web cumplen con las tres categorías de accesibilidad: estructurales, navegación y contenido alternativo. Por lo tanto, sería de tipo “Conformace Leve AA”, y podría llevar el logotipo indicado. Todas las imágenes incluyen sus correspondientes textos alternativos. No se proporcionan enlaces redundantes que lleven a otras páginas. En lo referente al color la página puede ser accedida sin el color. Y el texto es lo suficientemente claro para facilitar la interpretación.

Se hace un uso mínimo de tablas, solo con el perfil de administrador, por lo que la identificación de las cabeceras y los demás elementos de las mismas están muy acotados. Se ha probado a realizar una carga de la página sin ficheros CSS, y la carga, aunque lenta, es operativa y se puede leer e interpretar. La página es sencilla, la información es clara y entendible. Tampoco se han utilizado tecnologías adicionales como plug-ins o Adobe Flash.

### 3.2.3 Diseño del interfaz y contenidos de la página

Tiene una paleta de colores que resalta el cambio de información y una interfaz sencilla pero amigable, que cumple con el propósito de la página que es aportar información sobre cursos MOOC de interés para el usuario del buscador. Existe mucho espacio entre el texto que contiene la información expuesta. No existe sobresaturación, y los elementos de la página no desconciertan al usuario.

Se ha comprobado el resultado de la visualización los siguientes navegadores: Firefox, Chrome, Internet Explorer y Safari. La información expuesta es coherente en la exposición de la información. La redacción es corta y los textos no son demasiado largos, excepto en el campo información del curso, pero esta información proviene de los diferentes sitios analizados por la aplicación. Existen relaciones cruzadas entre los diferentes registros de las bases de datos implicadas en el proyecto, la exposición de estas relaciones es clara. Se evita lo máximo posible el uso de líneas horizontales para separar las cabeceras del texto.

## 3.3 Diseño de interfaces de usuario

En este apartado se muestra de forma detallada el diseño de la interfaz que será usada por los usuarios del sistema. Se ha procurado que dicha interfaz sea accesible para todos los usuarios, intentando hacer una interfaz basada en la usabilidad.

### 3.3.1 Home

Página de inicio para todos los usuarios. Es donde se dirige el servidor de aplicaciones en caso de que no se indique nada en la petición http. Desde esta página se puede realizar una búsqueda, ir a la página de registro, ir a la página de identificación o cambiar el idioma.

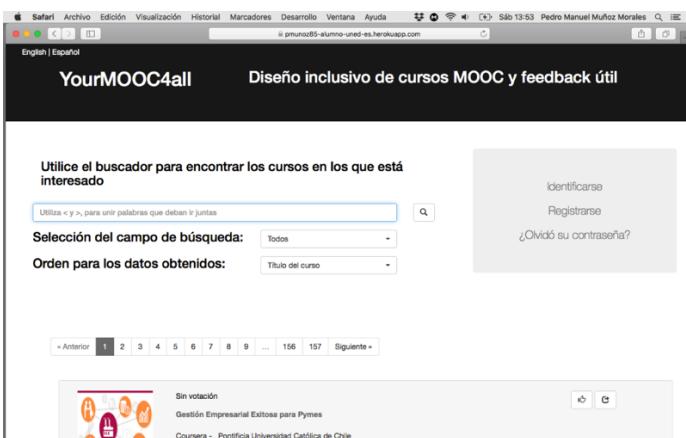


Figura 21 -Diseño - Home<sup>18</sup>

### 3.3.2 Administrar

Esta es la página de administración para los usuarios que tengan el rol de administrador, desde ella se puede gestionar todo lo relacionado con el sitio web. Gestionar cursos, usuarios, evaluaciones, lanzar tareas de búsqueda para nuevos cursos, etc.

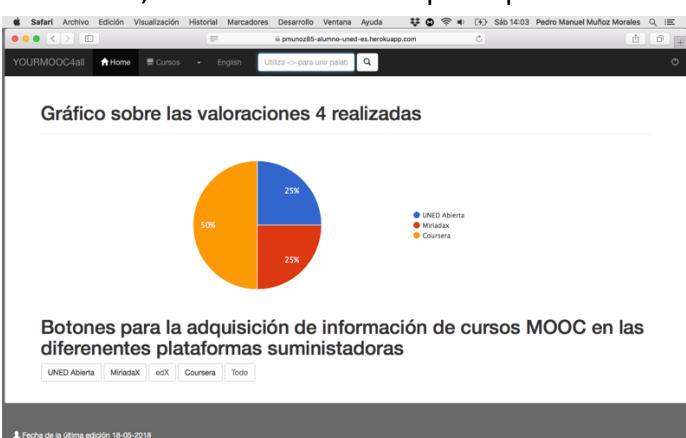


Figura 22 – Diseño - administrar<sup>19</sup>

### 3.3.3 Cursos

Aquí se pueden realizar todas las gestiones relacionadas con los cursos: como crear uno nuevo desde cero, modificar uno ya existente, borrar un curso, ver sus votaciones, marcarlo para que no se muestren en las búsquedas, etc.

<sup>18</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com>

<sup>19</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com/administrador/index?locale=es>

Figura 23 – Diseño - cursos<sup>20</sup>

### 3.3.4 Evaluación

En esta página se puede aportar la valoración sobre un curso. Esta valoración estará repartida en tres apartados diferentes: motivación, representación del contenido y expresión y acción a lo largo del curso, también se pueden añadir nuestros comentarios o modificarlos con posterioridad.

Figura 24 – Diseño - evaluación<sup>21</sup>

### 3.3.5 Inicio

Página de inicio para los usuarios registrados e identificados, en la que se observan los cursos que se han marcado como de interés en sesiones anteriores. También se pueden editar nuestros datos e incluso darnos de baja en el sistema.

<sup>20</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com/cursos?locale=es>

<sup>21</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com/evaluaciones/new?curso=34&locale=es>

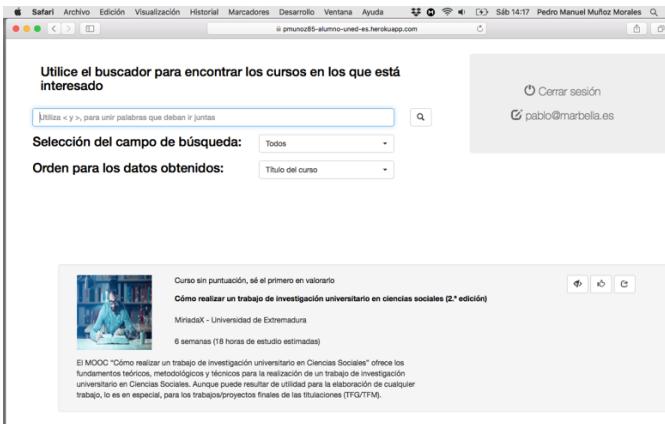


Figura 25 – Diseño – Inicio del usuario registrado<sup>22</sup>

### 3.3.6 Información, detalle de un curso

En esta página se puede ver la información de la que se dispone de un curso en particular, acceder a su votación o ver su puntuación obtenida.



Figura 26 – Diseño – información del curso<sup>23</sup>

## 3.4 Diagrama de navegación

A continuación, se muestra el diagrama de navegación por el que se moverá el usuario de la aplicación, mostrando la navegabilidad entre las distintas pantallas del sistema y como se relacionan unas páginas con otras.

Cada una de estas clases se corresponde con una interfaz gráfica, permitiendo la interacción con los usuarios del sistema. Cada interfaz gráfica está a su vez dividida en dos clases: el Controlador que hereda de la clase padre ApplicationController y el Modelo que hereda de la clase padre ActiveRecord. También en la parte de la Vista se encuentran los

<sup>22</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com/inicio/index?interesado=512&locale=es&muestra=1>

<sup>23</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com/cursos/512?locale=es>

ficheros con código HTML y Ruby, los cuales serán intercambiados por el contenido solicitado por el usuario.

Otras interfaces como Home, Inicio o Administración no contarán con la parte del Modelo, ya que no tabla en la base de datos del sistema. Del MVC, solo hacen uso de la Vista y el Controlador.

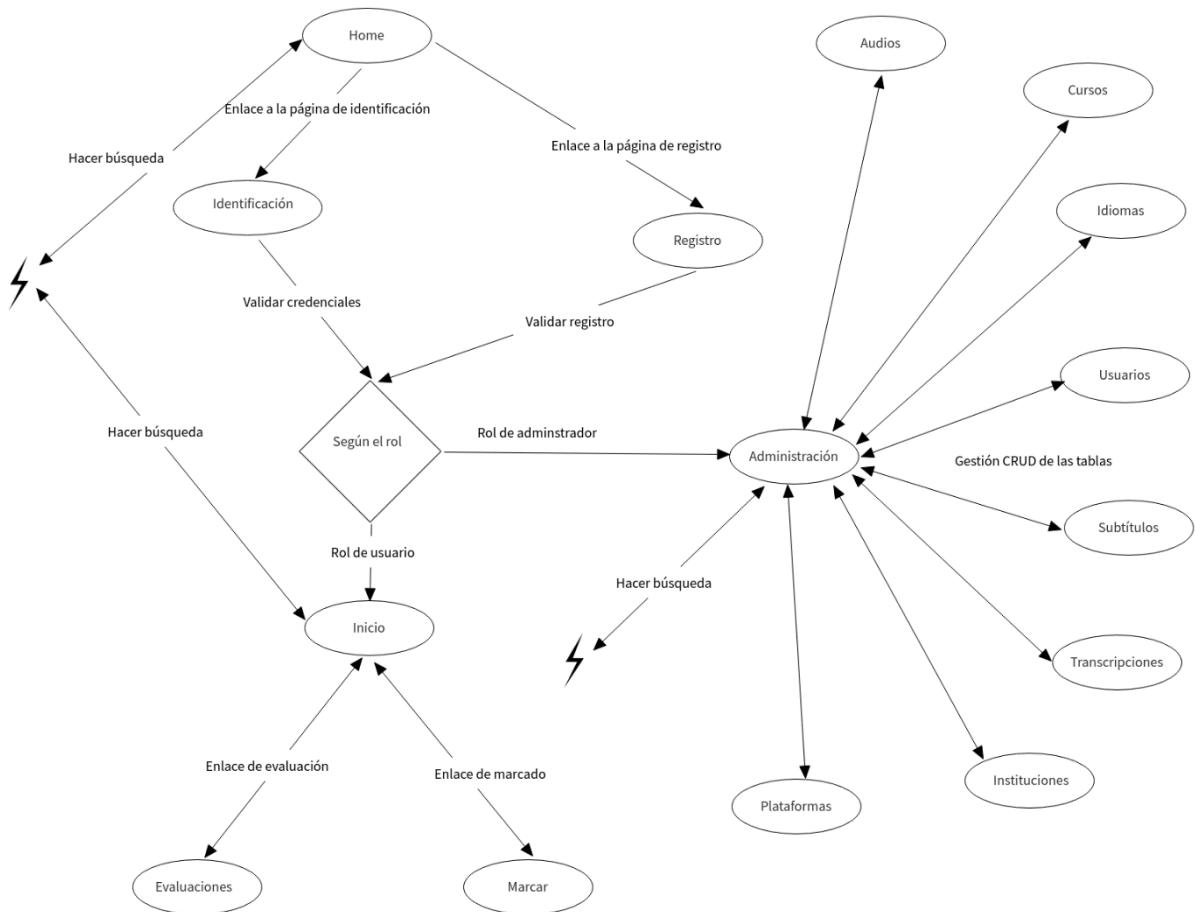


Figura 27 - Diagrama de navegación

### 3.5 Diagrama de clases

Los diagramas de clases muestran las clases de un sistema, sus atributos y las relaciones de herencia que existen entre ellas. La mayoría de estas clases corresponden al controlador de cada una de la Vistas que han sido definidas. Todas estas clases heredan de la clase padre ApplicationController y es donde se desarrolla la lógica de negocio.

Estas clases cuentan con métodos para la inserción de datos, la edición, el borrado y la muestra de información. En estos métodos se llenan de contenido las variables que más tarde se intercambiarán por información en las Vistas de cada una de las clases de la aplicación.

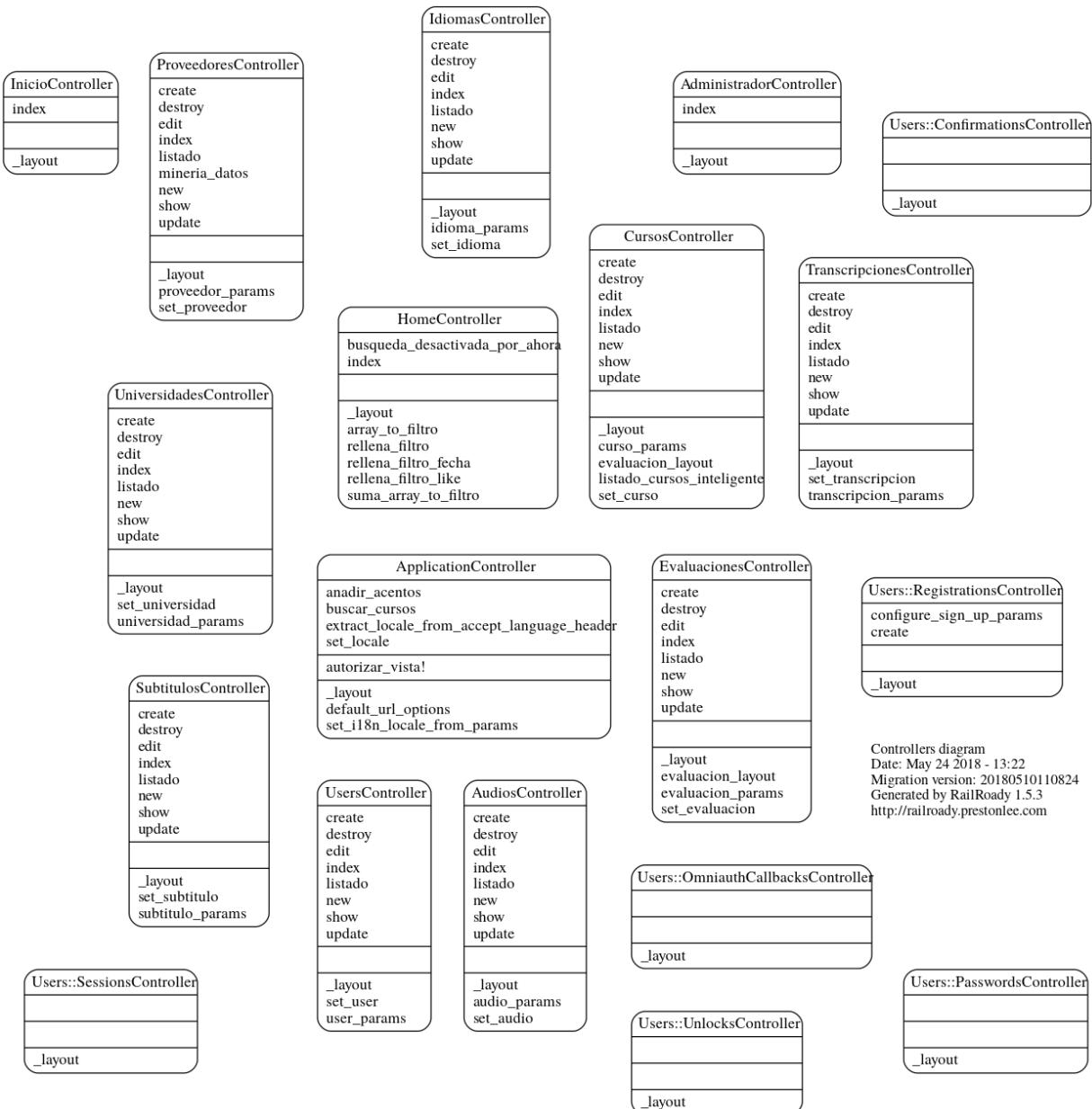
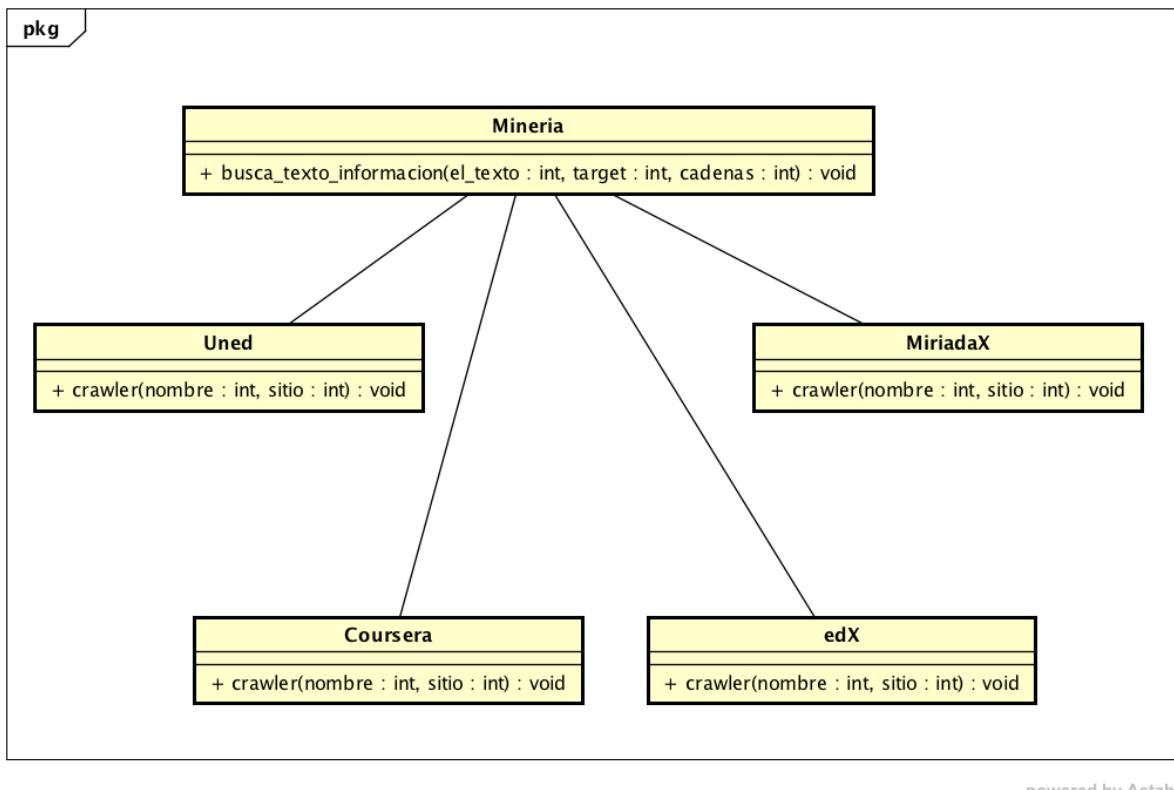


Figura 28 - Diagrama de clases

Hay acciones que cuentan con dos métodos para completar la acción, por ejemplo, la inserción de datos en un principio cuenta con el método new que muestra una vista con un formulario para la inserción de información, una vez que el usuario ha llenado de información dicho formulario y pulsa sobre el botón crear, se llama al segundo evento create, que es el encargado de guardar la información en la base de datos.

La clase Minería no hereda de ninguna otra clase, pero cuenta con cuatro clases hijas, que son: Uned, Miriadax, Coursera y Edx. En estas cuatro clases es donde se realiza toda la magia para adquirir la información desde las diferentes plataformas de forma automática y ordenada. Esta jerarquía de clases se ha creado así para poder compartir el método declarado en esta clase llamado busca\_texto\_informacion. De esta forma, al poner en marcha la búsqueda de

cuando en una plataforma en concreto se crea una instancia de la clase hija, por ejemplo, la clase Uned, y se llama a su método crawler, con los parámetros nombre y sitio.



powered by Astah

Figura 29 - Diagrama de minería

### 3.6 Diseño de la persistencia

La persistencia es la acción de preservar la información de un objeto de forma permanente, pero también se refiere a poder recuperar dicha información para que pueda ser utilizada nuevamente.

Las bases de datos elegidas han sido dos: para la parte de desarrollo y pruebas se ha decidido utilizar MySQL y para la parte de producción PostgreSQL, por imposición del servidor de aplicaciones en Internet. La aplicación desarrollada utiliza bases de datos relacionales para el nivel de persistencia, en base al propósito y necesidades de la aplicación. Es básico que la base de datos esté correctamente estructurada y diseñada con diversas entidades y sus correspondientes atributos relacionados entre sí, de forma coherente y eficaz.

### 3.6.1 Diagrama de Entidad Relación

En este apartado se puede observar el Diagrama de Entidad Relación a nivel de la base de datos, pero también el mismo diagrama visto desde la perspectiva del Modelo de bases de datos, esto es, la manera que la aplicación Ruby On Rails mapea las tablas y sus relaciones, pero desde el punto de vista aportado por ActiveRecord.

El primer diagrama contiene una visión de las tablas reales y los campos que se almacenan de manera permanente, eso no significa que sean los únicos, también existen campos calculados, pero estos no se almacenan en las bases de datos, sino que son calculados en tiempo de ejecución por el modelo de datos heredado de ActiveRecord.

En el proyecto hay varios campos calculados, pero sobre todo en el modelo Curso, porque este modelo contiene los cálculos de las puntuaciones recibidas por los usuarios registrados, y durante la representación de esa información en pantalla debe hacer uso de esos campos calculados, como, por ejemplo: mostrar\_puntuacion, puntuación y puntuación\_estrella, que son métodos de la clase Curos diseñados para tal fin.

## Diagrama Entidad Relación a nivel de la base de datos.

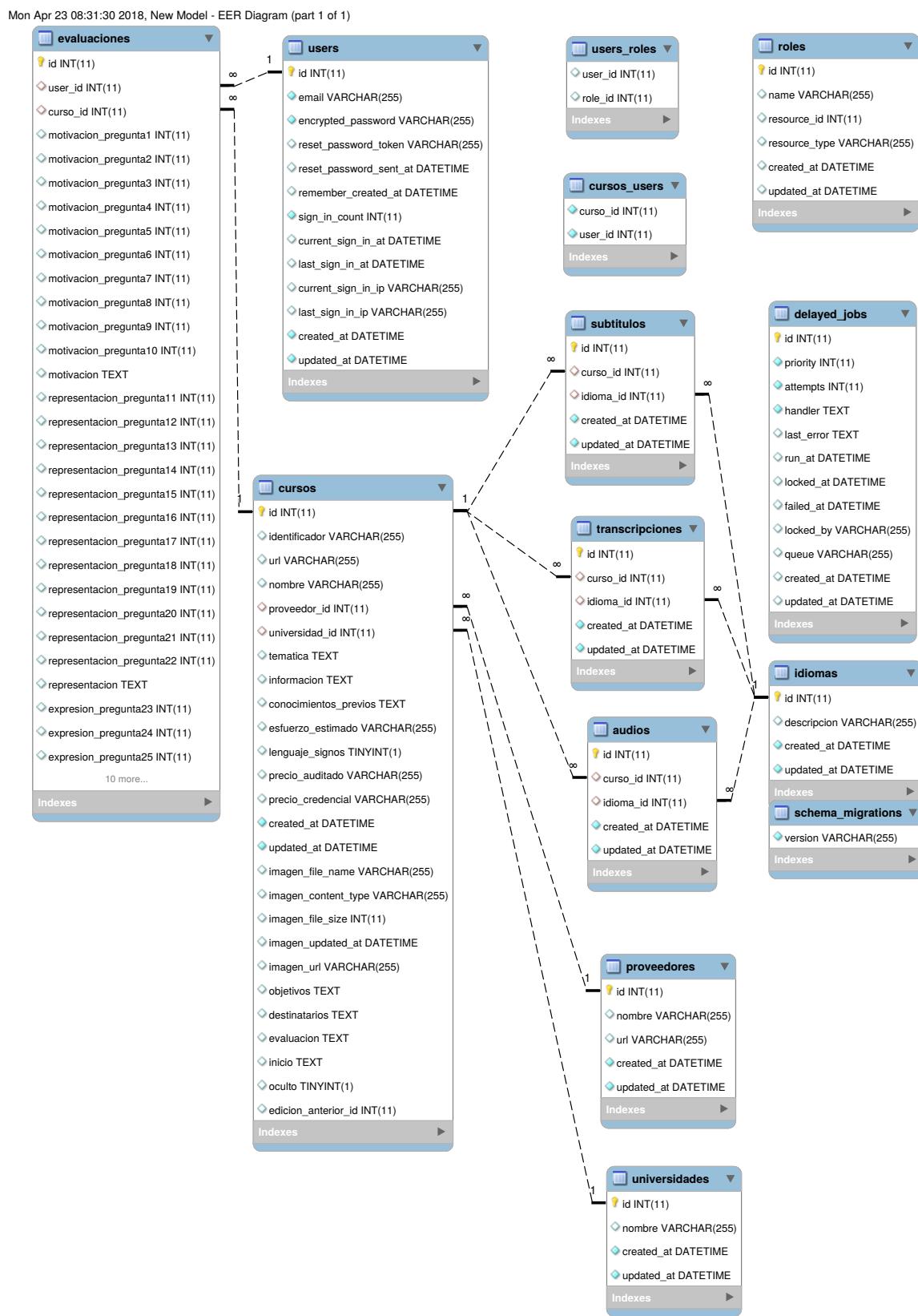


Figura 30 - Diagrama entidad relación

En este diagrama sencillo, a nivel de Modelo de datos, se pueden observar las relaciones existentes entre los modelos que representan las tablas de la base de datos, esta información es completa ya cuenta con las relaciones a nivel de base de datos y las relaciones establecidas a nivel de aplicación, controlada en estos casos por los modelos herederos de ActiveRecord.

Models diagram

Date: May 24 2018 - 13:22

Migration version: 20180510110824

Generated by RailRoady 1.5.3

<http://railroadylee.prestonlee.com>

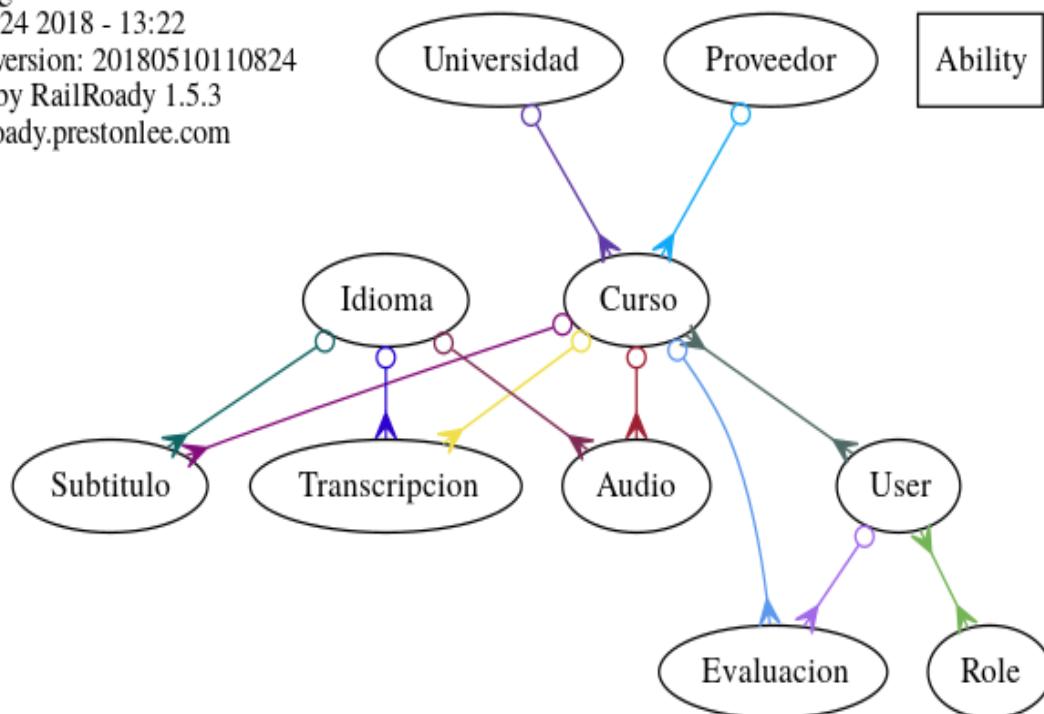


Figura 31 - Diagrama del modelo

Por último, se muestran las relaciones y los campos vistos desde los diferentes modelos existentes en el proyecto. Se debe destacar que los campos aquí representados son una abstracción de los campos que realmente están guardados en la base de datos, esto es así porque, en realidad no se trabaja directamente con el servidor de bases de datos, sino que se realiza a través de del ORM ActiveRecord.

Este detalle que en principio parece algo sin importancia, pero no lo es, ya que utilizar esta abstracción permite utilizar el mismo código para diferentes bases de datos, o al menos esa es la teoría. Esto hace posible disponer de un servidor de bases de datos para el desarrollo, en el caso MySQL, y otro servidor de bases de datos para la puesta en producción, en el caso PostgreSQL. Sin embargo, ha sido necesario habilitar diferente código para operar sobre cada una de ellas, lo que ha dificultado el desarrollo.

## Diagrama del Modelo de datos ActiveRecord

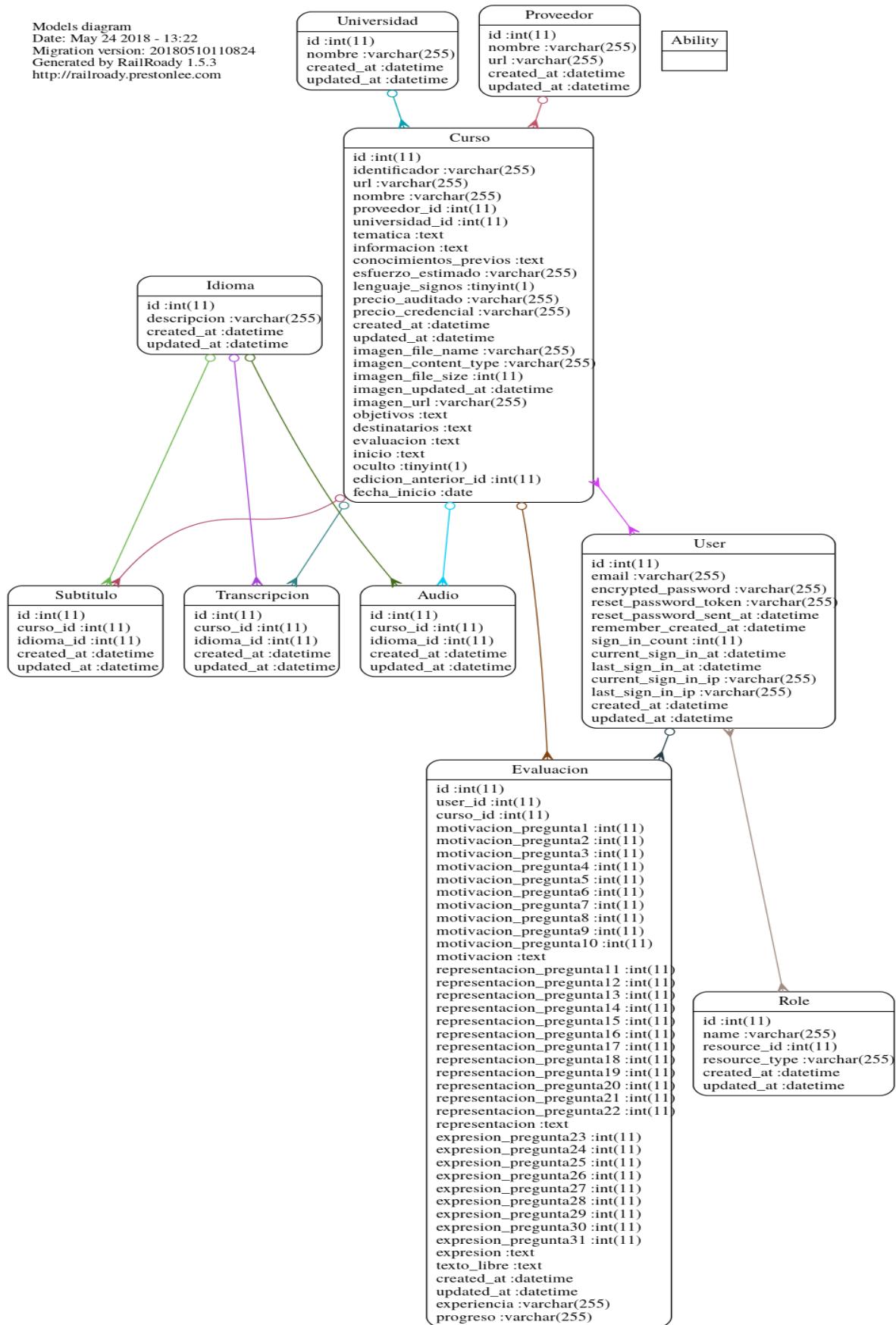


Figura 32 - Diagrama ActiveRecord

# 4. Implementación

En este apartado se va a describir los aspectos más interesantes de la fase de implementación del proyecto, además de comentar los problemas y contingencias surgidas durante el proceso de desarrollo del mismo. El código fuente al completo del proyecto se puede consultar en el repositorio GitHub<sup>24</sup>.

## 4.1 Partes fundamentales

El proyecto está dividido en tres partes fundamentales que son: la adquisición de información sobre cursos MOOC en plataformas web que ofrecen estos cursos, la búsqueda de cursos en las bases de datos del sistema que coincidan con los criterios de los usuarios y la evaluación de esos cursos por parte de los usuarios registrados en el sistema.

### 4.1.1 Adquisición de información o Harvesting

En esta parte del desarrollo se implementan las clases necesarias para extraer información de las plataformas web que ofrecen cursos MOOC, analizando las etiquetas HTML para identificar la información que se necesita en cada ámbito de información.

El procedimiento seguido para la extracción de información es el siguiente:

1. Se consigue el listado de cursos desde la plataforma oferente, partiendo de una URL ya conocida.
2. Se identifica cada curso por su etiqueta HTML.
3. Se extrae la URL del curso en particular y se accede a los detalles del curso.
4. Se identifica por sus etiquetas HTML los diferentes contenidos para la base de datos, como por son: El título del curso, quien lo imparte, cuanto tiempo dura, etc.
5. Antes de proceder a guardar datos que están relacionados en otras tablas, como por ejemplo que institución imparte el curso, se consulta su existencia y en caso de que no exista se crea una nueva institución, para así poder guardar su referencia en el curso que se está creado en este momento.
6. Se vuelve al punto 2 mientras queden enlaces de cursos por visitar.
7. Cuando se ha terminado con todos los enlaces a los cursos MOOC que existen en la página actual especificada en el punto 1, se avanza a la siguiente página de la plataforma.
8. Cuando se ha terminado con la última página de cursos que son ofrecidos por esta plataforma, se da por concluido el proceso de adquisición de información en este sitio web.

---

<sup>24</sup> <https://github.com/pmunoz85/YourMOOC4all>

A continuación, se muestra el código de adquisición para la plataforma Coursera, con la clase Minería y la clase Coursera la cual hereda de Minería.

```
1 # -*- encoding : utf-8 -*-
2 class Mineria
3
4     def busca_texto_informacion(el_texto, target, cadenas)
5
6         texto_en_plano = []
7         el_texto.each do |linea|
8             texto_en_plano << linea
9         end
10        texto_en_plano = texto_en_plano.join
11
12        h = {}
13        cadenas.each do |c|
14            if texto_en_plano.index(c)
15                h[c] = texto_en_plano.index(c)
16            end
17        end
18        h = h.sort_by { |key, value| value}.to_h
19
20        primero = true
21        segundo = true
22        final = ''
23        h.each do |key, value|
24            if primero then
25                if key == target then
26                    primero = false
27                end
28            else
29                if segundo then
30                    segundo = false
31                    final = key
32                end
33            end
34        end
35
36        objetivos = ''
37        if texto_en_plano.index/#{target}/)
38
39            objetivos = texto_en_plano[/#{target}(.*)#{final}/m, 1].strip
40        #    objetivos = texto_en_plano.scan(/#{target}(.*)#{final}/m).join
41        end
42        return objetivos
43    end
44 end
```

Figura 33 - Clase Minería

```

1 # -*- encoding : utf-8 -*-
2 ▼ class Coursera < Minería
3
4   def crawler(nombre, sitio)
5
6   ▼ # proveedor = Proveedor.find_by_nombre(nombre)
7
8     tag_apartado = '/courses?_facet_changed_=true&primaryLanguages=es'
9     tag_lista_cursos = 'a.rc-OfferingCard'
10    tag_nombre = '.title'
11    tag_imagen = 'img.offering-image'
12    tag_universidad = '.creator-names'
13    tag_tematica = '.course-description'
14    tag_informacion = '.content-inner'
15
16    tag_destinatarios = '.target-audience-section'
17    tag_tabla = '.basic-info-table'
18    tag_inicio = '.rc-StartDateString' #startdate rc-StartDateString caption-text
19
20    doc = Nokogiri::HTML(open(sitio + tag_apartado), nil, Encoding::UTF_8.to_s)
21
22    cursos = doc.css(tag_lista_cursos)
23
24    mas_paginas = true
25    pagina = 0
26    ▼ while mas_paginas do
27      cursos.each do |curso|
28        v_identificador = curso['href']
29
30        if Curso.find_by_identificador(v_identificador).nil? then
31          registro = Curso.new
32
33          # URL DEL CURSO
34          direccion_curso = sitio + v_identificador
35          detalle_curso = Nokogiri::HTML(open(direccion_curso), nil, Encoding::UTF_8.to_s)
36
37          # ID DEL CURSO
38          registro.identificador = v_identificador
39
40          # DIRECCIÓN DE ANÁLISIS
41          registro.url = sitio
42
43          # NOMBRE DEL CURSO
44          if detalle_curso.css(tag_nombre).text.empty? then
45            registro.nombre = detalle_curso.css('.s12n-headline').text
46          else
47
48            registro.nombre = detalle_curso.css(tag_nombre).text
49          end
50
51          # IMAGEN DEL CURSO
52          url_imagen = curso.css(tag_imagen)[0]['src']
53          url_imagen = url_imagen[0,url_imagen.index(/\?/)]
54          unless url_imagen.index(/http/).nil? then
55            registro.imagen_url = url_imagen
56          end
57
58          # PROVEEDOR
59          proveedor = Proveedor.find_by_nombre(nombre)
60          if proveedor.nil? then
61            proveedor = Proveedor.create(:nombre => nombre, :url => sitio)
62          end
63          registro.proveedor = proveedor
64
65          # UNIVERSIDAD
66          if detalle_curso.css(tag_universidad).text.sub(/Created by:/,'').empty? then
67            nombre_uni = detalle_curso.css('p.partner-marketing-blurb').text.strip[0,254]
68          else
69            nombre_uni = detalle_curso.css(tag_universidad).text.sub(/Created by:/,'')
70          end
71          id_uni = Universidad.find_by_nombre(nombre_uni)
72          if id_uni.nil? then
73            id_uni = Universidad.create(:nombre => nombre_uni)
74          end
75          registro.universidad = id_uni
76

```

Figura 34 - Clase Coursera

```

77      # TEMÁTICA pendiente
78      if detalle_curso.css(tag_tematica).text.nil? then
79          registro.tematica = detalle_curso.css('.s12n-subheader')#.text
80          tmp = ''
81      else
82          if detalle_curso.css(tag_tematica).length > 500
83              registro.tematica = detalle_curso.css(tag_tematica)#[0,500] + '....'#.text
84              registro.tematica = registro.tematica.gsub(/\s+/, '')
85              registro.tematica = registro.tematica.gsub(<script[^>]*>(.*)</script>,'')
86              registro.tematica = registro.tematica.gsub(<(.*)>,'')
87              tmp = registro.tematica[0,500]#.text
88              registro.tematica = tmp + '....'#.text
89              tmp = tmp.text
90      else
91          tmp = detalle_curso.css(tag_tematica)#.text
92          registro.tematica = tmp#.text
93          registro.tematica = registro.tematica.gsub(/\s+/, '')
94          registro.tematica = registro.tematica.gsub(<script[^>]*>(.*)</script>,'')
95          registro.tematica = registro.tematica.gsub(<(.*)>,'')
96          tmp = tmp.text
97      end
98  end
99
100 # INFORMACIÓN DEL CURSO
101 registro.informacion = detalle_curso.css(tag_informacion)
102 registro.informacion = registro.informacion.gsub(/^\s+/, '')
103 registro.informacion = registro.informacion.gsub(<script[^>]*>(.*)</script>,'')
104 registro.informacion = registro.informacion.gsub(<(.*)>,'')
105 registro.informacion = registro.informacion.gsub(tmp,'')
106
107 #Quitamos el 'About this course: '
108 registro.tematica = registro.tematica.gsub('About this course: ','')
109
110 # ESFUERZO ESTIMADO
111 esfuerzo = ''
112 # CONOCIMIENTOS PREVIOS
113 requerimientos = ''
114 # TIPOS DE ACTIVIDADES Y EVALUACION
115 evaluacion = ''
116 # OBJETIVOS DE APRENDIZAJE
117 objetivos = ''
118 detalle_curso.css(tag_tabla).css('tr').each do |linea|
119
120     if linea.text.index(/Commitment/)
121         esfuerzo = linea.text.gsub(/Commitment/, '')
122     end
123
124     if linea.text.index(/Hardware Req/)
125         requerimientos = linea.text.gsub(/Hardware Req/, '')
126     end
127
128     if linea.text.index(/How To Pass/)
129         evaluacion = linea.text.gsub(/How To Pass/, '')
130     end
131
132     if linea.text.index(/Level/)
133         objetivos = linea.text.gsub(/Level/, '')
134     end
135 end
136 registro.esfuerzo_estimado = esfuerzo
137 registro.conocimientos_previos = requerimientos
138 registro.evalucion = evaluacion
139 registro.objetivos = objetivos
140
141 # DESTINATARIOS Y NIVEL
142 registro.destinatarios = detalle_curso.css(tag_destinatarios).text
143
144 # FECHA DE INICIO
145 registro.inicio = detalle_curso.css(tag_inicio).text
146
147 # PRECIO AUDITADO
148 # PRECIO CREDITO
149 precio_auditado = ''
150 precio_credencial = ''
151
152 registro.save
153
154 end
155
156 pagina = pagina + 20
157 siguiente = sitio + '/courses?_facet_changed_=true&primaryLanguages=es&start=' + pagina.to_s
158 doc = Nokogiri::HTML(open(siguiente), nil, Encoding::UTF_8.to_s)
159 cursos = doc.css(tag_lista_cursos)
160
161 #mas_paginas = false if cursos.nil?
162 mas_paginas = false if cursos.count == 0
163
164 end
165 end
166
167 handle_asynchronously :crawler
168 end

```

#### 4.1.2 Búsqueda de cursos MOOC

En esta parte del proyecto es donde tiene lugar la magia para encontrar los cursos que el usuario está buscando. En el método “buscar\_cursos” se recoge, mediante parámetros de instancia, los valores de la etiqueta donde el usuario ha introducido el literal de búsqueda, el orden que el usuario ha seleccionado para los resultados obtenidos, y el campo sobre el que se realiza la búsqueda.

```
20  def buscar_cursos(ver_ocultos = false)
21    session[:campo_busqueda] = params[:campo_busqueda]
22    session[:orden_busqueda] = params[:orden_busqueda]
23
24    orden = 'cursos.nombre'
25    orden = 'cursos.nombre' if params[:orden_busqueda] == t("mio.orden_busqueda_titulo") #'Título del curso'
26    orden = 'universidades.nombre' if params[:orden_busqueda] == t("mio.orden_busqueda_instituciones") #'Instituciones'
27    orden = 'proveedores.nombre' if params[:orden_busqueda] == t("mio.orden_busqueda_plataformas") #'Plataformas'
28    orden = 'puntuacion' if params[:orden_busqueda] == t("mio.orden_busqueda_puntuacion") #'Puntuación obtenida'
29
30    if params[:buscar].nil? or params[:buscar].empty?
31      if orden == 'puntuacion' then
32        todos_cursos = Curso.joins(
33          'left join universidades on universidades.id = cursos.universidad_id
34          left join proveedores on proveedores.id = cursos.proveedor_id
35          ').sort_by(&:puntuacion) #.paginate(:per_page => 4, :page => params[:page])
36
37    cursos_array = []
38
39    todos_cursos.each do |t|
40      cursos_array << t.id
41    end
42
43    cursos_literal = cursos_array.join(', ')
44
45    if 'development' == ENV['RAILS_ENV'] then
46      @cursos = Curso.order("field(id, #{cursos_literal})").paginate(:per_page => 4, :page => params[:page])
47    else
48      @cursos = Curso.order_by_ids(cursos_array).paginate(:per_page => 4, :page => params[:page])
49    end
50  else
51    @cursos = Curso.joins(
52      'left join universidades on universidades.id = cursos.universidad_id
53      left join proveedores on proveedores.id = cursos.proveedor_id
54      ').order(orden).paginate(:per_page => 4, :page => params[:page])
55  end
56
57  return
58 end
59
60 @resultado = []
61 filtro = []
62 contenido = []
63 @buscar = params[:buscar]
64 @buscar_cursos = @buscar.split(' ').join('|')
65
66 palabras_juntas = @buscar.scan( /<(![\>])*>/)
67
68 palabras_sueltas = @buscar.gsub( /<(![\>])*>/,'').split(' ')
69
70 palabras_juntas.each do |palabra|
71   contenido << palabra.to_s.gsub(/["\`\\]/,'')
72 end
73
74 palabras_sueltas.each do |palabra|
75   contenido << palabra.to_s
76 end
77
78 linea = []
79 linea2 = []
80
81 #lenguaje_signos
82 #precio_auditado
83 #precio_credencial
84
85 if params[:campo_busqueda] == t("mio.campo_busqueda.todos") then #'Todos' then
86   contenido.each do |bus|
87     if 'development' == ENV['RAILS_ENV'] then
88       linea << "("
89       #{if not ver_ocultos then '(oculto IS NULL OR oculto IS FALSE) and ' end}
90       (upper(cursos.identificador) like upper(?)
91       or upper(cursos.nombre) like upper(?)
92       or upper(proveedores.nombre) like upper(?)
93       or upper(universidades.nombre) like upper(?)
94       or upper(cursos.tematica) like upper(?)
```

Figura 35 - Método buscar\_cursos

```

94         or upper(cursos.informacion) like upper(?)
95         or upper(cursos.conocimientos_previos) like upper(?)
96         or upper(cursos.esfuerzo_estimado) like upper(?)
97         or upper(cursos.objetivos) like upper(?)
98         or upper(cursos.destinatarios) like upper(?)
99         or upper(cursos.evalucion) like upper(?)
100        or upper(cursos.inicio) like upper(?)) ) "
101    else
102      linea << "(#if not ver_ocultos then (oculto IS NULL OR oculto IS FALSE) and ' end"
103      (translate(upper(cursos.identificador),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU'))
104      or translate(upper(cursos.nombre),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
105      or translate(upper(proveedores.nombre),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
106      or translate(upper(universidades.nombre),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
107      or translate(upper(cursos.tematica),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
108      or translate(upper(cursos.informacion),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
109      or translate(upper(cursos.conocimientos_previos),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
110      or translate(upper(cursos.esfuerzo_estimado),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
111      or translate(upper(cursos.objetivos),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
112      or translate(upper(cursos.destinatarios),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
113      or translate(upper(cursos.evalucion),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')
114      or translate(upper(cursos.inicio),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')) ) "
115    end
116
117    (1..12).each do
118      linea2 << "%#{bus}%""
119    end
120
121    @resaltado << bus
122  end
123  else
124    if not ver_ocultos then
125      oculto = '(oculto IS NULL OR oculto IS FALSE) and '
126    else
127      oculto = ''
128    end
129
130  case params[:campo_busqueda]
131    when t("mio.campo_busqueda_titulo") # "Título del curso"
132      nombre_campo = 'cursos.nombre'
133    when t("mio.campo_busqueda_tematica") # "Temática"
134      nombre_campo = 'cursos.tematica'
135    when t("mio.campo_busqueda_informacion") # "Información"
136      nombre_campo = 'cursos.informacion'
137    else
138      nombre_campo = 'cursos.nombre'
139    end
140
141
142    contenido.each do |bus|
143      if 'development' == ENV['RAILS_ENV'] then
144        linea << "( #{oculto} (upper(#{nombre_campo}) like upper(?)) ) "
145      else
146        linea << "( #{oculto} (translate(upper(#{nombre_campo}),'ÁÉÍÓÚ','AEIOU') like translate(upper(?),'ÁÉÍÓÚ','AEIOU')) ) "
147      end
148
149      linea2 << "%#{bus}%""
150
151      @resaltado << bus
152    end
153
154  end
155
156  filtro << linea.join(' and ')
157  linea2.each do |l|
158    filtro << l
159  end
160
161 filtro.join(', ')
162
163 if orden == 'puntuacion' then
164   todos_cursos = Curso.joins(
165     'left join universidades on universidades.id = cursos.universidad_id
166     left join proveedores on proveedores.id = cursos.proveedor_id
167     ').where(filtro).sort_by(&:puntuacion).paginate(:per_page => 4, :page => params[:page])
168
169 cursos_array = []
170 todos_cursos.each do |t|
171   cursos_array << t.id
172 end
173
174 cursos_literal = cursos_array.join(',')
175
176 if 'development' == ENV['RAILS_ENV'] then
177   @cursos = Curso.where("id in (#{{cursos_literal}})").order("field (id, #{{cursos_literal}})").paginate(:per_page => 4, :page => params[:page])
178 else
179   @cursos = Curso.where("id in (#{{cursos_literal}})").order_by_ids(cursos_array).paginate(:per_page => 4, :page => params[:page])
180 end
181
182 else
183   @cursos = Curso.joins(
184     'left join universidades on universidades.id = cursos.universidad_id
185     left join proveedores on proveedores.id = cursos.proveedor_id
186     ').where(filtro).order(orden).paginate(:per_page => 4, :page => params[:page])
187 end
188
189 @resaltado = anadir_acentos(@resaltado)
190
191 rescue_from CanCan::AccessDenied do |exception|
192   redirect_to root_url, :alert => exception.message
193 end
194
195 def anadir_acentos(resaltado)
196   return "" if resaltado.nil?
197
198   r = []
199
200   resaltado.each do |i|
201     i = i.gsub(/[\N\ñ]/,'[\N\ñ]')
202     i = i.gsub(/[\á\Á]/,'[\á\Á]')
203     i = i.gsub(/[\é\É]/,'[\é\É]')
204     i = i.gsub(/[\í\Í]/,'[\í\Í]')
205     i = i.gsub(/[\ó\Ó]/,'[\ó\Ó]')
206     i = i.gsub(/[\ö\Ö]/,'[\ö\Ö]')
207     i = i.gsub(/[\ú\Ü]/,'[\ú\Ü]')
208     r << i
209   end
210
211   return r
212 end

```

En el código se puede observar como se realizan consultas SQL diferentes para los ambientes de desarrollo y producción. Esto es debido a la utilización de dos servidores de datos diferentes: MySQL y PostgreSQL.

#### 4.1.3 Evaluación de los cursos

Otra parte importante es la retroalimentación de la información relacionada con la calidad de los cursos ofrecidos por las plataformas. Esto se puede realizar gracias a los usuarios registrados que aportan su experiencia sobre cursos que están cursando o han cursado. Esas puntuaciones sumadas y después divididas para hallar la media sobre cinco puntos.

Esta evaluación del curso está dividida en tres apartados: la motivación, la representación y la expresión. La puntuación es mostrada en varios formatos, como la puntuación en forma de estrellas o la puntuación mostrada como texto. Las puntuaciones son compartidas entre las diferentes ediciones que pudiera tener un curso, por ejemplo, si un curso de matemáticas tiene 3 ediciones diferentes, estas tendrán la misma puntuación. El código necesario para conseguir esto está repartido entre el modelo Evaluación y el modelo Curso. El modelo Curso contiene los métodos necesarios para calcular la puntuación de un usuario sobre un curso y el modelo Evaluación contiene los métodos para calcular la media de todas las puntuaciones de un curso en concreto.

Solo se muestra el código del método puntuación, que hace referencia a la puntuación obtenida por un curso de todos los usuarios registros que han valorado este curso.

```
137  def puntuacion
138    suma = 0.0
139
140    cuantos = 0
141
142    tmp = self
143    ant = self
144
145    unless tmp.edicion_anterior_id.nil? then
146      tmp = Curso.find(tmp.edicion_anterior_id)
147      while not tmp.nil? and not tmp.edicion_anterior_id.nil?
148        if tmp.id == self.id then
149          tmp = nil
150        else
151          if tmp.edicion_anterior_id.nil?
152            ant = tmp
153            tmp = nil
154          else
155            tmp = Curso.find(tmp.edicion_anterior_id)
156          end
157        end
158      end
159    end
160
161    while not ant.nil?
162      ant.evaluaciones.each do |e|
163        suma = suma + e.media
164        cuantos = cuantos + 1
165      end
166
167      ant = Curso.find_by_edicion_anterior_id(ant.id)
168
169      unless ant.nil? then
170        if ant.id == self.id then
171          ant = nil
172        end
173      end
174    end
175
176    if cuantos == 0 then
177      return 0
178    else
179      return -1 * (suma / cuantos)
180    end
181  end
```

Figura 36 - Método puntuación

## 4.2 Esquema de la base de datos

A continuación, se detallan un par de tablas utilizadas en este proyecto, así como sus relaciones entre ellas, tipo de campo, longitud del mismo, si tiene valor por defecto y si cuenta con alguna restricción. Para aportar toda esta información detallada sobre cada tabla de la base de datos, se muestra la instrucción de creación en SQL de cada elemento.

```
25 CREATE TABLE `cursos` (
26   `id` int(11) NOT NULL AUTO_INCREMENT,
27   `identificador` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
28   `url` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
29   `nombre` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
30   `proveedor_id` int(11) DEFAULT NULL,
31   `universidad_id` int(11) DEFAULT NULL,
32   `tematica` text COLLATE utf8_unicode_ci,
33   `informacion` text COLLATE utf8_unicode_ci,
34   `conocimientos_previos` text COLLATE utf8_unicode_ci,
35   `esfuerzo_estimado` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
36   `lenguaje_signos` tinyint(1) DEFAULT NULL,
37   `precio_auditado` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
38   `precio_credencial` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
39   `created_at` datetime NOT NULL,
40   `updated_at` datetime NOT NULL,
41   `imagen_file_name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
42   `imagen_content_type` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
43   `imagen_file_size` int(11) DEFAULT NULL,
44   `imagen_updated_at` datetime DEFAULT NULL,
45   `imagen_url` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
46   `objetivos` text COLLATE utf8_unicode_ci,
47   `destinatarios` text COLLATE utf8_unicode_ci,
48   `evaluacion` text COLLATE utf8_unicode_ci,
49   `inicio` text COLLATE utf8_unicode_ci,
50   `oculto` tinyint(1) DEFAULT NULL,
51   `edicion_anterior_id` int(11) DEFAULT NULL,
52   `fecha_inicio` date DEFAULT NULL,
53 ) ENGINE=InnoDB AUTO_INCREMENT=2179 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
54 KEY `index_cursos_on_proveedor_id` (`proveedor_id`) USING BTREE,
55 KEY `index_cursos_on_universidad_id` (`universidad_id`) USING BTREE,
56 KEY `index_cursos_on_edicion_anterior_id` (`edicion_anterior_id`),
57 CONSTRAINT `fk_rails_48779263f2` FOREIGN KEY (`proveedor_id`) REFERENCES `proveedores` (`id`),
58 CONSTRAINT `fk_rails_91c896c6e6` FOREIGN KEY (`universidad_id`) REFERENCES `universidades` (`id`)
59 ) ENGINE=InnoDB AUTO_INCREMENT=2179 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

Figura 37 – esquema- tabla cursos

```
25 CREATE TABLE `evaluaciones` (
26   `id` int(11) NOT NULL AUTO_INCREMENT,
27   `user_id` int(11) DEFAULT NULL,
28   `curso_id` int(11) DEFAULT NULL,
29   `motivacion_pregunta1` int(11) DEFAULT NULL,
30   `motivacion_pregunta2` int(11) DEFAULT NULL,
31   `motivacion_pregunta3` int(11) DEFAULT NULL,
32   `motivacion_pregunta4` int(11) DEFAULT NULL,
33   `motivacion_pregunta5` int(11) DEFAULT NULL,
34   `motivacion_pregunta6` int(11) DEFAULT NULL,
35   `motivacion_pregunta7` int(11) DEFAULT NULL,
36   `motivacion_pregunta8` int(11) DEFAULT NULL,
37   `motivacion_pregunta9` int(11) DEFAULT NULL,
38   `motivacion_pregunta10` int(11) DEFAULT NULL,
39   `motivacion` text COLLATE utf8_unicode_ci,
40   `representacion_pregunta11` int(11) DEFAULT NULL,
41   `representacion_pregunta12` int(11) DEFAULT NULL,
42   `representacion_pregunta13` int(11) DEFAULT NULL,
43   `representacion_pregunta14` int(11) DEFAULT NULL,
44   `representacion_pregunta15` int(11) DEFAULT NULL,
45   `representacion_pregunta16` int(11) DEFAULT NULL,
46   `representacion_pregunta17` int(11) DEFAULT NULL,
47   `representacion_pregunta18` int(11) DEFAULT NULL,
48   `representacion_pregunta19` int(11) DEFAULT NULL,
49   `representacion_pregunta20` int(11) DEFAULT NULL,
50   `representacion_pregunta21` int(11) DEFAULT NULL,
51   `representacion_pregunta22` int(11) DEFAULT NULL,
52   `representacion` text COLLATE utf8_unicode_ci,
53   `expresion_pregunta23` int(11) DEFAULT NULL,
54   `expresion_pregunta24` int(11) DEFAULT NULL,
55   `expresion_pregunta25` int(11) DEFAULT NULL,
56   `expresion_pregunta26` int(11) DEFAULT NULL,
57   `expresion_pregunta27` int(11) DEFAULT NULL,
58   `expresion_pregunta28` int(11) DEFAULT NULL,
59   `expresion_pregunta29` int(11) DEFAULT NULL,
60   `expresion_pregunta30` int(11) DEFAULT NULL,
61   `expresion_pregunta31` int(11) DEFAULT NULL,
62   `expresion` text COLLATE utf8_unicode_ci,
63   `texto_libre` text COLLATE utf8_unicode_ci,
64   `created_at` datetime NOT NULL,
65   `updated_at` datetime NOT NULL,
66   `experiencia` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
67   `progreso` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
68 ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
69 KEY `index_evaluaciones_on_user_id` (`user_id`),
70 KEY `index_evaluaciones_on_curso_id` (`curso_id`),
71 CONSTRAINT `fk_rails_55cac0be62` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`),
72 CONSTRAINT `fk_rails_8635f6e02a` FOREIGN KEY (`curso_id`) REFERENCES `cursos` (`id`)
73 ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

Figura 38 - esquema – tabla evaluaciones

## 4.3 Ejemplo de funcionamiento del paradigma MVC

Se va a ver un pequeño ejemplo con la interfaz Transcripciones:

En la parte correspondiente al Modelo, la clase Transcripción hereda de ActiveRecord. Se puede ver en dicho código que la tabla transcripciones cuenta con dos relaciones con otras tablas, que son cursos e idiomas. Cada transcripción tiene una referencia a un curso y otra a un idioma al que pertenece dicha transcripción. También se observa que tiene dos restricciones donde se indica que la transcripción en los campos curso\_id e idioma\_id no pueden tener un valor nulo.

```
1 class Transcripcion < ActiveRecord::Base
2   belongs_to :curso
3   belongs_to :idioma
4
5   validates :curso_id, :presence => true
6   validates :idioma_id, :presence => true
7
8   def curso_nombre
9     return self.curso.nil? ? "" : self.curso.nombre
10  end
11
12  def curso_nombre=(valor)
13    self.curso = Curso.find_by_nombre(valor)
14  end
15  def idioma_descripcion
16    return self.idioma.nil? ? "" : self.idioma.descripcion
17  end
18
19  def idioma_descripcion=(valor)
20    self.idioma = Idioma.find_by_descripcion(valor)
21  end
22
23  def to_s
24    "Transcripcion #{id}"
25  end
26
27 end
28
```

Figura 39 - ejemplo mvc 1

En la parte correspondiente al Controlador existe la clase TranscripcionesController que hereda de la clase padre ApplicationController. En ella se puede apreciar un método por cada evento que contiene la URL a la que se tiene acceso desde el enrutador.

```

1  # -*- encoding : utf-8 -*-
2  class TranscripcionesController < ApplicationController
3    before_filter :authenticate_user!
4    load_and_authorize_resource #skip_load_resource only: [:create]
5    skip_load_resource only: [:create]
6
7    before_action :set_transcripcion, only: [:show, :edit, :update, :destroy]
8
9    layout 'administrador'
10
11   def listado
12     datos = []
13     Transcripcion.all.each do |j|
14       datos << j.descripcion
15     end
16
17     render :json => datos.to_json
18   end
19
20
21   def index
22     @transcripciones = Transcripcion.all
23     @transcripciones = Transcripcion.paginate(:page => params[:page], :per_page => 20)
24   end
25
26   def show
27     # @registros = @transcripcion.RELACIONADOS.paginate(:page => params[:page], :per_page => 20)
28   end
29
30   def new
31     @transcripcion = Transcripcion.new(:curso_id => params['curso_id'])
32   end
33
34   def edit
35   end
36
37   def create
38     @transcripcion = Transcripcion.new(transcripcion_params)
39
40     if @transcripcion.save
41       redirect_to @transcripcion.curso, notice: t('mio.create_transcripcion')
42     else
43       render action: 'new'
44     end
45   end
46
47   def update
48     if @transcripcion.update(transcripcion_params)
49       redirect_to @transcripcion.curso, notice: t('mio.update_transcripcion')
50     render action: 'edit'
51   end
52 end
53
54   def destroy
55     @transcripcion.destroy
56     redirect_to transcripciones_url, notice: t('mio.destroy_transcripcion')
57   end
58
59   private
60     # Use callbacks to share common setup or constraints between actions.
61     def set_transcripcion
62       @transcripcion = Transcripcion.find(params[:id])
63     end
64
65     # Only allow a trusted parameter "white list" through.
66     def transcripcion_params
67       #params.require(:transcripcion).permit(:curso_id, :idioma_id)
68
69       params.require(:transcripcion).permit(:curso_descripcion, :curso_nombre, :curso_name, :idioma)
70     end
71   end

```

Figura 40 - ejemplo mvc 2

Cada Vista corresponde con un método del controlador y un fichero HTML con código Ruby, el cual será sustituido por el controlador. A continuación, se muestra el que pertenece al evento Show del controlador y está contenido en el fichero show.html.erb.

```
1 <%- model_class = Transcripcion -%>
2 <div class="col-md-12">
3   <div class="well col-md-12 ">
4     <dl class="dl-horizontal">
5       <dt>Curso:</dt>
6       <dd><%= @transcripcion.curso %></dd>
7       <dt>Idioma:</dt>
8       <dd><%= @transcripcion.idioma %></dd>
9
10    </dl>
11
12   <div class="panel-footer">
13     <%= link_to transcripciones_path, :class => 'btn btn-default' do %>
14       <span class="glyphicon glyphicon-list-alt"></span>
15       <%= t('.back', :default => t("helpers.links.back", :default => 'Volver')) %>
16     <% end %>
17     <%= link_to edit_transcripcion_path(@transcripcion), :class => 'btn btn-primary' do %>
18       <span class="glyphicon glyphicon-pencil"></span>
19       <%= t('.edit', :default => t("helpers.links.edit", :default => 'Editar')) %>
20     <% end %>
21
22     <%= link_to transcripcion_path(@transcripcion), :method => 'delete', :data => { :confirm =>
23       t('.confirm', :default => t("helpers.links.confirm", :default => "¿Eliminar el
24       transcripcion '#{@transcripcion.id}'?") ), :class => 'btn btn-danger' do %>
25       <span class="glyphicon glyphicon-trash"></span>
26       <%= t('.destroy', :default => t("helpers.links.destroy")) %>
27     <% end %>
28   </div>
29 </div>
30 </div>
```

Figura 41 - ejemplo mvc 3

## 4.4 Árbol de directorios de la aplicación

En esta sección se detalla la estructura de directorios del proyecto, mostrando el árbol de carpetas relevantes de la aplicación.

moocotor	drwxrwxr-x	13	usuario	usuario	4096	abr	16	09:30	.
	drwxr-xr-x	41	usuario	usuario	4096	may	6	19:52	..
	drwxrwxr-x	8	usuario	usuario	4096	feb	20	19:15	app
	-rw-rw-r--	1	usuario	usuario	232	mar	24	21:01	'apuntes para la memoria.txt'
	drwxrwxr-x	2	usuario	usuario	4096	mar	8	14:47	bin
	drwxrwxr-x	5	usuario	usuario	4096	abril	18	19:56	config
	-rw-rw-r--	1	usuario	usuario	153	feb	20	19:15	config.ru
	-rw-rw-r--	1	usuario	usuario	1974	abril	12	10:30	'Crear scaffold.txt'
	drwxrwxr-x	3	usuario	usuario	4096	abril	20	23:51	db
	-rw-rw-r--	1	usuario	usuario	2245	abril	20	09:55	Gemfile
	-rw-rw-r--	1	usuario	usuario	6585	abril	20	09:55	Gemfile.lock
	drwxrwxr-x	7	usuario	usuario	4096	mayo	19	23:32	.git
	-rw-rw-r--	1	usuario	usuario	427	mar	2	14:30	.gitignore
	drwxrwxr-x	6	usuario	usuario	4096	abril	17	14:50	lib
	drwxrwxr-x	2	usuario	usuario	4096	mayo	26	23:06	log
	-rw-r-----	1	usuario	usuario	37	mar	9	11:48	Procfile
	drwxrwxr-x	6	usuario	usuario	4096	mar	26	17:25	public
	-rw-rw-r--	1	usuario	usuario	249	febrero	20	19:15	Rakefile
	-rw-rw-r--	1	usuario	usuario	8049	abril	7	11:48	README.md
	-rw-rw-r--	1	usuario	usuario	478	febrero	20	19:15	README.rdoc
	drwxrwxr-x	8	usuario	usuario	4096	febrero	20	19:15	test
	drwxrwxr-x	6	usuario	usuario	4096	febrero	22	09:38	tmp
	-rw-rw-r--	1	usuario	usuario	22856	abril	12	19:23	traducción
	drwxrwxr-x	3	usuario	usuario	4096	febrero	20	19:15	vendor

*Figura 42 - árbol de directorios*

- App: En este directorio se organiza la estructura general de la aplicación. Contiene los subdirectorios que albergan las distintas capas del modelo MVC.
- App/assets: Este directorio contiene todo lo necesario para el front-end: JavaScript, CSS e imágenes, está todo unificado en este directorio porque con el comando “rake assets:precompile” se unifica todo en un mismo fichero, se le quitan los comentarios y los espacios para que se carguen más rápidamente en el cliente.
- App/controllers: Este directorio contiene todos los controladores de la aplicación, que se ocupan de gestionar la lógica de negocio para la aplicación.
- App/helpers: Este directorio contiene todos los ficheros, denominados helpers, que contiene código Ruby para ocultar la complejidad en la que no tendría que formar parte la vista, pero hay que recordar que en los helpers tampoco debería haber código HTML.
- App/mailers: Este directorio contiene todos los ficheros para manipular los correos electrónicos. Estos ficheros se generan con el comando “rails generate mailer nombreMailer”.
- App/models: Este directorio contiene todos los ficheros para mapear la base de datos, configurar las relaciones, añadir los campos calculados, añadir restricciones a la hora de crear los registros, etc.
- App/views: Este directorio contiene todo lo relacionado con las vistas. Contiene un directorio por cada vista de un controlador y un fichero o más por cada método del controlador. También están almacenados los “partials”, que son ficheros con partes de código HTML y Ruby para componer una vista completa. Otro directorio importante que está contenido aquí es el directorio “layouts”, donde se almacenan las cabeceras y los pies que van a compartir las vistas.
- Bin: Este directorio contiene todos los ejecutables que utiliza Rails: bundle, rails, rake, setup y spring.
- Config: Este directorio almacena todos los ficheros con las reglas de ejecución para la aplicación, en sus tres ambientes diferentes: development, test y production. También cuenta con ficheros de configuración para: las rutas, las bases de datos, los idiomas, los inicializadores, etc.
- Db: Este directorio almacena todos los ficheros que hacen referencia a la creación de la base de datos, la introducción de información con las semillas, las migraciones y el esquema actual de la base de datos.
- Lib: Este directorio contiene todos los ficheros relacionados con la meta-programación, las tareas y las plantillas. La meta-programación es otro de los puntos fuertes de Ruby On Rails, consistente en código que genera más código.
- Log: Este directorio contiene todos los ficheros log para cada uno de los tres ambientes de ejecución: development, test y production.
- Public: Este directorio contiene los ficheros que serán accesibles desde el exterior, contiene los ficheros estáticos y los assets compilados.
- Test: Este directorio contiene todos los ficheros que hacen referencia a las pruebas realizadas con Minitest. En nuestro caso todos los ficheros aquí presentes han sido generados de forma automática por Ruby On Rails.

- Tmp: Este directorio contiene todos los ficheros temporales necesarios para el funcionamiento de la aplicación, como por ejemplo la cache y los ficheros de sesión.
- Vendor: Este directorio contiene los ficheros con el código de terceros en forma de librerías y plugins.
- /: Este directorio contiene algunos ficheros de configuración que no están en la carpeta config, como el Gemfile que sirve para indicar que gemas necesita la aplicación para funcionar. También cuenta con el fichero config.ru, que contiene la configuración Rack para iniciar la aplicación. Por último, se cuenta con el fichero Rakefile el cual localiza y carga las tareas que pueden ser ejecutadas desde la línea de comandos.

## 4.5 Control de versiones

Los sistemas de control de versiones sirven para controlar los cambios realizados durante el desarrollo de software, o cualquier actividad informática en que haya ficheros y vayan a sufrir muchos cambios durante un tiempo. Estas herramientas permiten conocer el estado actual un proyecto, unir los cambios desarrollados por diferentes personas, ver los cambios sufridos en cualquier fichero del proyecto, quien realizó los cambios, deshacer los cambios, etc.

Existen dos tipos de control de versiones:

1. Los sistemas centralizados, en estos casos existe un servidor centralizado el cual se encarga de mantener el repositorio y los desarrolladores tienen en su poder solo los ficheros en los que van a trabajar. Solo existe un lugar, el servidor, en el cual se encuentra todo el proyecto completo. Algunos ejemplos de este tipo de control de versiones son: Subversión y CVS.
2. Los sistemas distribuidos, en estos casos cada desarrollador tiene un repositorio local completo, esto permite al desarrollador ir haciendo copias de su trabajo sin necesidad de estar conectado a ningún servidor. A la hora de compartir el trabajo realizado se puede optar por hacerlo por parejas, equipos o por todos los integrantes a la vez, no existe una única forma de realizarlo, es mucho más flexible.

### 4.5.1 Git

Para el control de versiones en este proyecto se ha utilizado la herramienta Git, un control de versiones distribuido diseñado por Linus Torvalds. Algunas de las características más importantes son: rapidez en la gestión de ramas, gestión distribuida, gestión eficiente de proyectos grandes y re-almacenamiento periódico de paquetes.

Aunque un proyecto de desarrollo cuente con un solo programador, es recomendable usar un gestor de versiones, ya que esta herramienta no solo permite unir el código desarrollado en paralelo por los programadores, sino que además te permite hacer ramas para tener

diferentes versiones de nuestro desarrollo, volver a un punto anterior de un fichero en concreto, revisar los cambios que ya has realizado consultado el log de la herramienta, clonar tu desarrollo en el servidor, etc. Git distribuye a cada programador una copia local del historial del desarrollo completo, y los cambios se propagan entre los repositorios locales. Estos cambios se incorporan como ramas adicionales y pueden ser fusionados de la misma manera que se hace con las ramas locales. Los repositorios pueden publicarse por http, FTP o SSH. Cuenta con una gestión muy eficiente de proyectos grandes, dada la rapidez de gestión de diferencias entre archivos.

Este es el listado completo de commits que se han realizado durante todo el proyecto, son puntos de restauración del proyecto que se guardan como copias de seguridad en una línea temporal:

```
* 7a941fb (HEAD, origin/master, heroku/master, master) Reparación de un error con el parámetro interesado
* 71d97dd Retiramos los enlaces del cambio de idioma duplicados
* 1b15da7 Traducidas las etiquetas y los desplegables de las evaluaciones
* 46e302c Otro ajuste más para el error en la adquisición de datos en Coursera
* f8cbc14 Reparado un error en la adquisición de datos en Coursera
* 8b698ff Dos traducciones rezagadas
* 111a273 Más cambios para la traducción del sitio web
* 50802b5 Reparado el error del controlador de los subtítulos
* f11b76d Cambios en las etiquetas de los idiomas
* a146fba Cambios solicitados por Paco
* 0c448e9 Cambios solicitados por Paco en su correo electrónico de este miércoles
* 2db020d Reparado un error con las ediciones anteriores
* 62facbf Cambio de algunas etiquetas y reparación de los duplicados en UNED
* f35fe38 rake assets:precompile
* 30a691e Aplicación de las mejoras visuales para el contraste y la adicción de un selector para el campo de búsqueda
* 943f8db Varios ajustes pequeños
* 459ad92 Primer intento de reparación en el group by en Postgresql
* 3fd9712 Realización de las sugerencias aportadas por Covadonga y Paco
* 45250f0 Mostrar sólo las preguntas de evaluación y un más para el resto del texto
* 051834b Tercera prueba para PostgreSQL
* bf56bb5b Segunda prueba para PostgreSQL
* 489477e Primer intento para hacer lo mismo pero en PostgreSQL
* c53d718 Añadimos el orden de los resultados por puntuación, válido para MySql
* afbc6f9 Realización de los puntos resultantes de la reunión con Covadonga y Paco.
* fa1798b Recuperamos el orden de prelación cuando el usuario no está logueado
* 78af700 Modificamos la información del host para el envío de correos desde Heroku
* 8f6d1b9 Corregido el error del texto resaltado
* f6fb32f Ajustes visuales sobre los Show
* e762600 Corregido el error del nombre de usuario
* 10357df Nuevo ajustes visuales para la puntuación
* bd267f2 Volvemos a activar Delayed_job para Heroku, porque me he dado cuenta de que el nombre del método llamado no era el correcto
* 5f1bee9 No funciona esta línea handle_asynchronously :crawler_web en Heroku
* 759236f He vuelto a añadir la línea handle_asynchronously :crawler_web, para probarla otra vez en Heroku
* 5e85148 Añadimos un namespace a la tarea, para ver si funciona en Heroku
* f5c9f03 Añadida una tarea para realizar el análisis de las webs. No funciona en Heroku porque quiere una tarjeta de crédito
* 5955d11 Añadimos la funcionalidad de cambiar el orden de los resultados de las búsquedas
* 68f3f53 Corregido el error de despliegable inteligente
* 923f4cc Añadida la funcionalidad de compartir las puntuaciones entre diferentes ediciones de un mismo curso.
* 594d396 Modificamos el aspecto del paginado
* 4bb168e Mejoras para la visualización en un teléfono
* c9ff523 Modificadas las migraciones para evitar el error de Heroku
* aade943 Añadida la funcionalidad: estoy interesado en el curso
* b3ddb06 Añadimos los comentarios limitado a 5 en las búsquedas
* 7bb3f3f Deshacemos los cambio de la activación de Delayed_job por problemas en Heroku
* 88644c6 Activamos el delayed_job otra vez
* 550d313 Tercer intento de reparación del error de los acentos
* 845218e Segundo intento de Reparación del error de los acentos y el color en la búsquedas
* 495af92 Reparación del error de los acentos y el color en la búsquedas
* a532389 Devolvemos los cambios para probar la llamada sin parámetro
* 134715e Borrado del fichero custom_failure.rb del directorio lib
* 03eed54 Cambio para Heroku, porque lo que hicimos anteriormente no funciona allí
* 43c5eeef Solucionado el cambio de orden cuando esta autorizado el usuario, pide que se registre
* 91e4977 Reparada la visualización de los botones cuando se ve en el teléfono.
```

Captura de pantalla...19.02.10

```

* 95ec0ce Corregido 2 errores de visualización
* 345cd72 Fecha de actualización y pluralize para las puntuaciones
* a9f8160 Añadimos un gráfico a la puntuación, para que resulte más agradable a la vista
* 4b7b1c4 Cambios estéticos y funcionales de las votaciones
* 915667a Iniciamos el acercamiento a la puntuación de los cursos MOOC
* 7b8bef7 Más cambios para los dos idiomas
* 5bc65a3 Primera prueba para soportar multiples idiomas.
* 71f1cd8 Más ajustes visuales para Devise
* 12a5a73 Ajustes visuales para Devise
* 83f4fb1 Cambiamos la estética de todo lo referente a Devise
* e3f2bd5 rake assets:precompile para Heroku
* 77b8a3c Añadimos los subtítulos, los idiomas y las transcripciones a los cursos.
* 8ff28bc Quitamos más gemas para reparar la vulnerabilidad
* d776cd0 Actualizamos Ruby On Rails a la versión 4.2.10
* aaf047d Añadimos el README para GitHub y lo subimos a la plataforma
* 6a5f258 rake assets:precompile para Heroku
* 073f936 Segundas mejoras para la accesibilidad desde la página web de examiner.ws
* 335a795 Primeras mejoras para la accesibilidad desde la página web de examiner.ws
* f2f6482 Cambiar la fecha de la Última edición
* 00b8e7c Modificadas todas las vistas para dale algo de color a la aplicación
* 5ab480d Segundo intento de diseño en la página home, usuario sin registrar.
* bf1f10f Primer intento de solucionar el problema de las búsquedas en Postgresql
* 4c6b726 Primer diseño de la página de inicio, home, para los usuarios no identificados.
* 87b3b31 Solucionado el error del proveedor y sustituidas las etiquetas de Proveedor por Plataforma y Universidad por Institución.
* dc3951c Borrados los campos que sobran de proveedores
* d7d872e Añadido el análisis de la web de Coursera
* 55f6836 Solucionado el problema de la mayúsculas y minúsculas en Postgresql

* 7e5e6e6 Parece que está solucionado el ERROR de la entrada en partes del programas sin autorización.
* f11bc38 Hemos añadido las búsquedas en los tres perfiles, pero me he dado cuenta de que no funciona bien el control de roles.
* 9396d03 Más ajustes para UNED Abierta.
* 8f1bb6f Terminada la parte de adquisición de datos desde la web de la UNED.
* 53182a2 Solucionado el problemas con las tildes y las ñ
* 07a934a Hacemos un commit para subir los cambios a Heroku.
* a53ed70 Hemos cambiado la forma de analizar las distintas web, ahora empleamos las etiquetas en el código, no en la base de datos y hemos hecho un fichero para cada web.
* 851d175 Reparado la etiqueta de MiriadaX y he vuelto a usar delayed_job
* 18450ff Volvemos atrás para ver en Heroku la razón de que no acepte los cursos de MiriadaX
* bcb0f80 Modificamos el fichero Procfile, para ver si es posible ejecutar los trabajos retrasados de forma automática.
* 0a09c7a He quitado la linea del redirect_to, me falta añadir alguna forma de indicar que el proceso ha terminado
* c3bd19d Añadimos la adquisición de datos en una clase independiente y la gestionamos con la gema delayed_job
* 10ca18e Reparación de la etiqueta de información al adquirir información
* f61b9f0 Borrada una alución que había al name de usuario.
* 89693a5 Añadimos la gema bootstrap-select para proveedor y universidad.
* c5bee34 Borrando restos en el código defectuoso
* 3d7b8bd Borrando restos en el código del campo identificador_curo
* d00de97 Reparado el nombre_cuso
* 3f75c77 Borrado del usuario invitado en el fichero seeds.rb
* 1d10247 Cambio de paparclip por url de imágenes del cursos, también hemos borrado 2 campos que no usamos, name en users y identificador_curso de proveedores.
* 98a9d65 Solucionado el error en la configuración
* f012ed3 Hemos traducido todo el entorno Devise al castellano, he borrado todas la alusiones al campo name de la tabla users, solo queda borrar el campo y también he añadido la configuración para el envío de emails.
* d904d16 Descartado un temporal.

* 333517a Funciona la adquisición de UNED Abierta y MiriadaX
* cef0cec Especificamos la version 0.20.0 de la gema pg para Heroku
* ffad0b4 Especificamos la version 0.21.0 de la gema pg
* d23e074 Hemos añadido modificaciones en el Gemfile para Heroku.
* 924f651 Modificación del Gemfile para bajar la gema postgress
* e0cc795 cambiamos la contraseña y el usuario de la base de datos en Heroku
* f2a3de0 Más cambios para Heroku
* 7fdac51 Modificación de la gema postgres en heroku
* 65d97a8 Configuración de Heroku
* b994330 Mejoras en la adquisición de datos
* caf8e0e Primer acercamiento a la adquisición de información automática.
* 56a13e8 Cambio de uso para el home, inicio y administrador.
* d9b9be9 Primer acercamiento a la adquisición de datos desde la web de UNED Abierta.
* 2509513 añadir usuarios online, registros, rails generate devise:controllers users
* 63ccfe1 Bodados los 2 temporales
* 4971de4 Parece que el sistema de control de usuarios va más o menos bien.
* b18599a Primeros pasos en la estructura del sitio web
* 7f91fe2 Inicio de la lógica de negocio.
* db5f256 Inicio

```

Figura 43 - commits de Git

#### 4.5.2 GitHub

Es un servicio para el alojamiento de repositorios de software gestionados por el sistema de control de versiones Git. Este servicio está pensado para hacer posible el compartir el código de manera fácil a través de Internet. Este servicio es gratuito solo si el código se distribuye de forma pública o, como en el caso de estudiantes universitarios. Si no se cumplen estas condiciones hay que pagar por el alojamiento del software de forma privada.

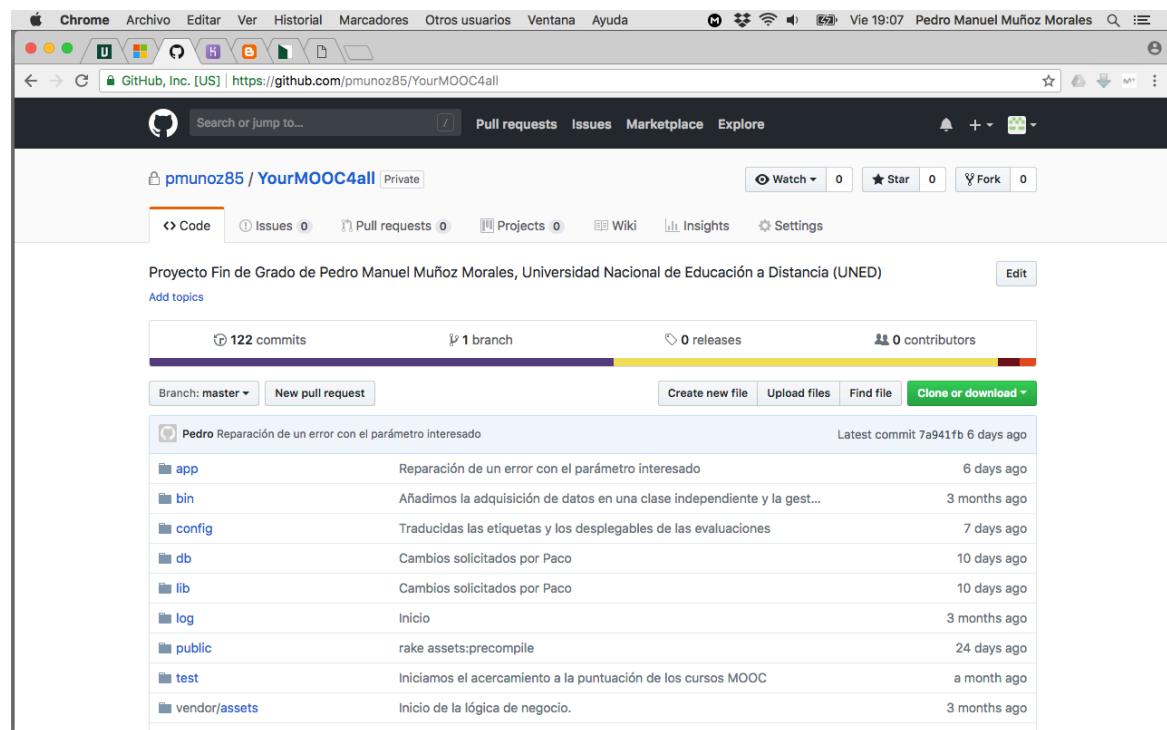


Figura 44 – GitHub<sup>25</sup>

## 4.6 Alojamiento en la web

Para servir la aplicación web en Internet se puede optar por montar nuestro propio servidor físico dedicado, por un servidor virtual propio, un servidor compartido, virtual o físico, o por último alojarlo en un Hosting en el que se comparten los procesadores, sistemas de bases de datos y ancho de banda.

En este proyecto la decisión ha sido alojarlo en Heroku por su facilidad de uso, su fiabilidad, gestión web y sobre todo por su coste 0, siempre que no se exceda de 500 MiB en la base de datos. Las dos ventajas más llamativas del alojamiento web en Heroku son: la visión online de los ficheros log y el acceso a la consola desde el navegador web. Heroku es un servicio de Hosting en la nube que soporta varios lenguajes. Inicialmente fue desarrollada en 2007 para soportar aplicaciones desarrolladas en Ruby, pero posteriormente se ha extendido a Java, Node.js, Python, Clojure, Scala, etc. Las aplicaciones se ejecutan desde un servidor

<sup>25</sup> <https://github.com/pmunoz85/YourMOOC4all>

usando DNS para apuntar al dominio de la aplicación<sup>26</sup>. El servidor Git de Heroku maneja los repositorios de las aplicaciones subidas por los usuarios.

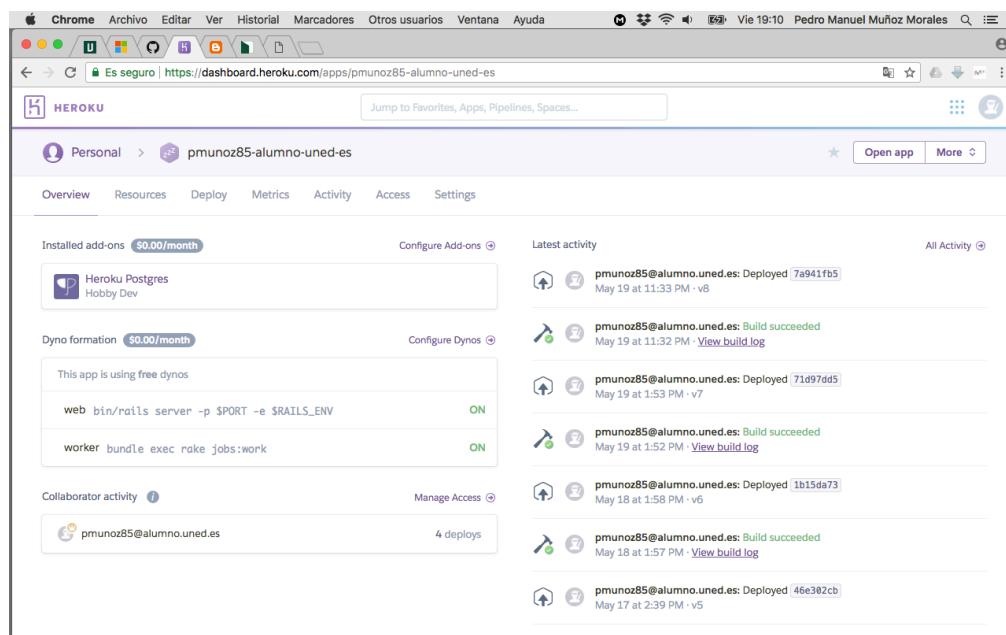


Figura 45 - Captura de pantalla con la cuenta de alojamiento en Heroku.

Heroku se basa en los denominados Dynos, que son piezas fundamentales de su arquitectura. Estas piezas proveen de capacidad de cómputo dentro de la plataforma: cada Dynos está aislado, por lo que su funcionamiento no afecta al resto y aporta funcionalidades a las aplicaciones para ser ejecutadas.

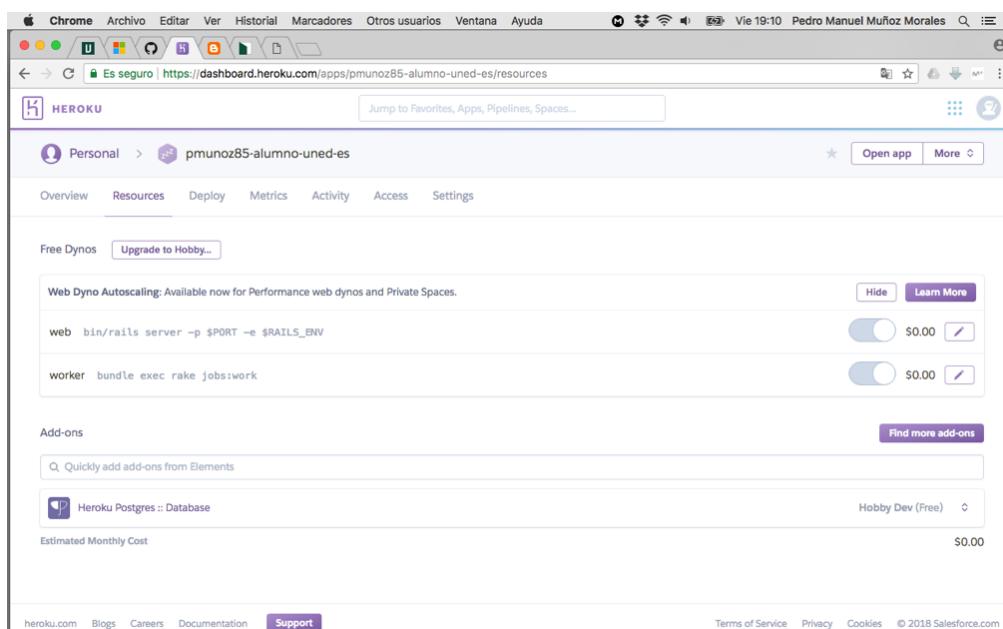


Figura 46 – Configuración de Heroku.

<sup>26</sup> <https://pmunoz85-alumno-uned-es.herokuapp.com>

## Principales características del Hosting:

- **Elasticidad y crecimiento.** La cantidad de Dynos asignados a una aplicación se puede cambiar en cualquier momento a través de la línea de comandos o el dashboard.
- **Tamaño.** Heroku ofrece diferentes tipos de dynos, cada uno con diferentes capacidades de procesamiento y memoria.
- **Routing.** Internamente los routers realizan un seguimiento de la ubicación de los Dynos que estén corriendo, y redirigen el tráfico de acuerdo con la misma.
- **Seguimiento.** Existe un manejador de Dynos, el cual monitorea de forma continua los Dynos que se estén ejecutando. En caso de una falla en un Dynos, este es eliminado y creado nuevamente.
- **Distribución y redundancia.** Los Dynos se encuentran aislados uno de otro. Esto implica que, de existir fallos en la infraestructura interna de alguno de ellos, los otros Dynos no se ven afectados, y consecuentemente tampoco la aplicación.

The screenshot shows the Heroku Dashboard interface. At the top, there's a navigation bar with links for Archivo, Editar, Ver, Historial, Marcadores, Otros usuarios, Ventana, Ayuda, and user information (Vie 19:11 Pedro Manuel Muñoz Morales). Below the navigation is a search bar and a 'Jump to Favorites, Apps, Pipelines, Spaces...' button. The main area has a 'HEROKU' logo and a 'Personal' dropdown. The app name 'pmunoz85-alumno-uned-es' is displayed. A 'More' button is also present. Below this, a navigation menu offers Overview, Resources, Deploy, Metrics, Activity, Access, and Settings. The 'Overview' tab is selected. Under 'Application Logs', there's a large text area showing log entries from May 2018. The logs include timestamps, application names (app[web.1]), and detailed log messages such as database queries and file rendering times. A checkbox for 'Autoscroll with output' is checked at the bottom left of the log area, and a 'Save' button is at the bottom right. At the very bottom of the page, there are links for heroku.com, Blogs, Careers, Documentation, Support, Terms of Service, Privacy, Cookies, and a copyright notice for © 2018 Salesforce.com.

Figura 47 – Log de Heroku.

Desde la página web de gestión para tu aplicación web, existe un enlace al fichero log de la aplicación. Esto en principio no parece gran cosa, pero ha hecho la vida mucho más fácil a la hora de reparar errores de ejecución. Realiza un seguimiento en tiempo real de lo que está ocurriendo en el servidor y en caso de error, se ve un mensaje detallado de donde se ha producido y el porqué.

En definitiva, es uno de los mejores, por no decir el mejor, hosting que existe actualmente para los desarrollos de aplicaciones Ruby On Rails. Por su instalación simple, por ser gratuito para aplicaciones pequeñas y por su forma simple de hacer los despliegues mediante el comando push de Git.

## 4.7 Tarea diaria

Se ha implementado una tarea para la adquisición de datos desde las plataformas oferentes de cursos MOOC, esta tarea es automatizada, transparente al administrador y con un intervalo de 24 horas. Heroku provee de una funcionalidad para llevar a cabo esta tarea llamada Heroku Scheduler, pero esta funcionalidad tiene un coste y el desarrollo del proyecto se debe ajustar a la premisa inicial de usar Open Source y servicios online de coste cero. Para ello se ha creado una tarea en el desarrollo que se ejecutará con el comando “rake”, esta tarea se llama “cursos:update”.

```
rake cursos:update # Esta tarea es llamada desde Heroku scheduler add-on
```

A continuación, solo quedará crear otra tarea en cualquier dispositivo, utilizando por ejemplo el cron de Linux, y hacer una llamada a esta tarea en Heroku con el siguiente comando. El cual creará cuatro peticiones a Delayed\_job para realizar la búsqueda en las cuatro plataformas oferentes definidas.

```
MacBook-Pro-de-Pedro:mooctor admin$ heroku run rake cursos:update
Running rake cursos:update on ⬤ pmunoz85-alumno-uned-es... up, run.3511 (Free)
Actualizando cursos ...
... Comienza el análisis de UNED Abierta
  (0.9ms) BEGIN
  SQL (1.1ms) INSERT INTO "delayed_jobs" ("handler", "run_at", "created_at", "updated_at") VALUES ($1, $2, $3, $4)
  RETURNING "id" [[{"handler": "--- !ruby/object:Delayed::PerformableMethod\nobject: !ruby/object:Uned {}\\nmethod_name: :crawler_without_delay\\nargs:\\n- UNED Abierta\\n- https://iedra.uned.es\\n"}, {"run_at": "2018-06-10 08:50:39.180579"}, {"created_at": "2018-06-10 08:50:39.180926"}, {"updated_at": "2018-06-10 08:50:39.180926"}]]
  (1.7ms) COMMIT
... Comienza el análisis de Miriadax
  (0.8ms) BEGIN
  SQL (1.9ms) INSERT INTO "delayed_jobs" ("handler", "run_at", "created_at", "updated_at") VALUES ($1, $2, $3, $4)
  RETURNING "id" [[{"handler": "--- !ruby/object:Delayed::PerformableMethod\\nobject: !ruby/object:Miriadax {}\\nmethod_name: :crawler_without_delay\\nargs:\\n- Miriadax\\n- https://miriadax.net\\n"}, {"run_at": "2018-06-10 08:50:39.198728"}, {"created_at": "2018-06-10 08:50:39.199216"}, {"updated_at": "2018-06-10 08:50:39.199216"}]]
  (3.3ms) COMMIT
... Comienza el análisis de Coursera
  (1.7ms) BEGIN
  SQL (1.5ms) INSERT INTO "delayed_jobs" ("handler", "run_at", "created_at", "updated_at") VALUES ($1, $2, $3, $4)
  RETURNING "id" [[{"handler": "--- !ruby/object:Delayed::PerformableMethod\\nobject: !ruby/object:Coursera {}\\nmethod_name: :crawler_without_delay\\nargs:\\n- Coursera\\n- https://www.coursera.org\\n"}, {"run_at": "2018-06-10 08:50:39.222153"}, {"created_at": "2018-06-10 08:50:39.222474"}, {"updated_at": "2018-06-10 08:50:39.222474"}]]
  (1.5ms) COMMIT
... Comienza el análisis de edx
Actualización terminada.
```

Figura 48 - tarea

Este es el código que contiene la tarea a la que se llama, si se estuviera interesado en abonar el precio para activar el Heroku Scheduler, solo se tendría que indicar que esta tarea

debe ser ejecutada cada 24 horas y no sería necesario implementar esta llamada desde otro terminal como aquí se indica.

```
1 |namespace :cursos do
2 |
3 |desc "Esta tarea es llamada desde Heroku scheduler add-on"
4 |task :update => :environment do
5 |  puts "Actualizando cursos ..."
6 |
7 |  puts "... Comienza el análisis de UNED Abierta"
8 |  w = Uned.new
9 |  w.crawler('UNED Abierta', 'https://iedra.uned.es')
10 |
11 |  puts "... Comienza el análisis de Miriadax"
12 |  w = Miriadax.new
13 |  w.crawler('MiriadaX', 'https://miriadax.net')
14 |
15 |  puts "... Comienza el análisis de Coursera"
16 |  w = Coursera.new
17 |  w.crawler('Coursera', 'https://www.coursera.org')
18 |
19 |  puts "... Comienza el análisis de edX"
20 |  w = Edx.new
21 |  w.crawler('edX', 'https://www.edx.org')
22 |
23 |  puts "Actualización terminada."
24 |
25 |
26 |end
27 |
```

Figura 49 - código tarea

## 5. Pruebas

La fase de pruebas es la parte más importante de todo el desarrollo, ya que es donde se verifica que el producto desarrollado tiene validez y cumpliendo con lo especificado en el apartado de análisis. Es tan importante que cada vez más desarrollos están dirigidos por las pruebas, los famosos TDD, Test-Driven Development.

Un producto que no sea robusto o que no cumpla lo especificado en el análisis es un producto fracasado, porque el usuario no lo usará por miedo al mal funcionamiento o que no le encuentra utilidad ya que no es la herramienta que él solicitó.

Para cumplir con este objetivo se han realizado diferentes pruebas:

- Las pruebas unitarias permiten comprobar el correcto funcionamiento de un módulo de código, así se puede comprobar que los módulos funcionan correctamente de una manera individual. Esto no permite localizar los errores de una forma óptima, si el proyecto no estuviera dividido en módulos, localizar un error entre miles de líneas de código sería una tarea imposible de realizar.
- Las pruebas de integración son el siguiente paso natural de las pruebas, una vez que se comprueba el funcionamiento de los módulos por separado, se comprueban que funcionan de manera conjunta. Se logra este objetivo uniendo módulos desde la raíz hacia las ramas o, al contrario, desde las ramas hasta la raíz, finalizando el proceso cuando esté el proyecto unido por completo.
- Las pruebas funcionales están basadas en la ejecución, la revisión y la retroalimentación de las funcionalidades diseñadas para este proyecto.
- Las pruebas de usabilidad son las que se encargan de comprobar que la interacción entre el usuario y el software, estas pruebas se enfocan en medir la capacidad de una aplicación en cumplir el propósito para el que fue diseñada. Consisten en seleccionar a un grupo de usuarios y solicitarles que realicen las tareas para las cuales fue diseñada la aplicación, mientras que el equipo de desarrollo toma notas sobre la respuesta de los usuarios.
- Las pruebas de accesibilidad son un subgrupo de las pruebas de usabilidad en las que los usuarios tienen discapacidades que afectan a su manera de utilizarlo.
- Las pruebas de seguridad pretenden descubrir las vulnerabilidades que afectan al funcionamiento de una aplicación.
- Las pruebas de manuales son aquellas que se realizan de una forma manual pero sistemática.

## 5.1 Pruebas unitarias

El Framework Ruby On Rails cuenta con un directorio destinado a las pruebas. Cada vez que se crea un nuevo modelo, Rails genera un fichero dentro del directorio “./test/models” con el mismo nombre que el modelo creado.

Las pruebas unitarias concernientes al modelo son aquellas que se encuentran relacionadas con los datos de la aplicación, creación de registros, borrado de registros y actualizaciones. Estas pruebas van orientadas al correcto cumplimiento de normas, excepciones y relaciones establecidas en la base de datos, aquí se verifica que la información aportada no contradice ninguna validación. Se ha creado un directorio “./test/fixtures” para llenar con información las bases de datos. Esta información para el llenado de las bases de datos debe estar escrita en YAML y debe haber un fichero por cada modelo del proyecto.

Tareas Rake para la ejecución de las pruebas:

Tareas	Descripción
rake test	Ejecuta todas las pruebas en el directorio <code>test</code> . También puedes ejecutar <code>rake</code> y Rails ejecutará todos las pruebas por defecto
rake test:controllers	Ejecuta todas las pruebas de controlador desde <code>test/controllers</code>
rake test:functionals	Ejecuta todas las pruebas funcionales desde <code>test/controllers</code> , <code>test/mailers</code> , y <code>test/functional</code>
rake test:helpers	Ejecuta todas las pruebas de los helpers desde <code>test/helpers</code>
rake test:integration	Ejecuta tadas las pruebas de integración desde <code>test/integration</code>
rake test:jobs	Ejecuta todas las pruebas de trabajos desde <code>test/jobs</code>
rake test:mailers	Ejecuta todas las pruebas de envíos de correos desde <code>test/mailers</code>
rake test:models	Ejecuta todas las pruebas del modelo desde <code>test/models</code>
rake test:units	Ejecuta todas las pruebas unitarias desde <code>test/models</code> , <code>test/helpers</code> , y <code>test/unit</code>
rake test:db	Ejecuta todas las pruebas en el directorio <code>test</code> y resetea la base de datos

Tabla 16 - tareas rake

## 5.2 Pruebas funcionales

Las pruebas funcionales en Rails son para los controladores y consiste en validar las acciones que suelen ejecutar los controladores, normalmente son las peticiones que realiza el navegador web y llegan a la aplicación. Dicho de otra manera, las pruebas funcionales validan las acciones que realiza la aplicación cuando responden las peticiones entrantes.

En este caso también se ha creado un directorio donde Rails deposita ficheros con los nombres de los controladores que hay en la aplicación. Estos ficheros contienen todas las pruebas para los eventos estándar que proporcionan los controladores, si se hace necesario hacer más pruebas funcionales aquí sería en lugar natural para depositarlas.

```
1 require 'test_helper'
2
3 class CursosControllerTest < ActionController::TestCase
4   setup do
5     @curso = cursos(:one)
6   end
7
8   test "should get index" do
9     get :index
10    assert_response :success
11    assert_not_nil assigns(:cursos)
12  end
13
14  test "should get new" do
15    get :new
16    assert_response :success
17  end
18
19  test "should create curso" do
20    assert_difference('Curso.count') do
21      post :create, curso: { conocimientos_previos: @curso.conocimientos_previos,
22                           esfuerzo_estimado: @curso.esfuerzo_estimado, identificador: @curso.identificador, imagen:
23                           @curso.imagen, informacion: @curso.informacion, lenguaje_signos: @curso.lenguaje_signos,
24                           nombre: @curso.nombre, precio_auditado: @curso.precio_auditado, precio_credencial: @curso.
25                           precio_credencial, proveedor_id: @curso.proveedor_id, tematica: @curso.tematica,
26                           universidad_id: @curso.universidad_id, url: @curso.url }
27    end
28
29    assert_redirected_to curso_path(assigns(:curso))
30  end
31
32  test "should show curso" do
33    get :show, id: @curso
34
35    assert_response :success
36
37  test "should get edit" do
38    get :edit, id: @curso
39    assert_response :success
40  end
41
42  test "should update curso" do
43    patch :update, id: @curso, curso: { conocimientos_previos: @curso.conocimientos_previos,
44                                       esfuerzo_estimado: @curso.esfuerzo_estimado, identificador: @curso.identificador, imagen: @
45                                       curso.imagen, informacion: @curso.informacion, lenguaje_signos: @curso.lenguaje_signos,
46                                       nombre: @curso.nombre, precio_auditado: @curso.precio_auditado, precio_credencial: @curso.
47                                       precio_credencial, proveedor_id: @curso.proveedor_id, tematica: @curso.tematica,
48                                       universidad_id: @curso.universidad_id, url: @curso.url }
49    assert_redirected_to curso_path(assigns(:curso))
50  end
51
52  test "should destroy curso" do
53    assert_difference('Curso.count', -1) do
54      delete :destroy, id: @curso
55    end
56
57    assert_redirected_to cursos_path
58  end
59 end
```

Figura 50 - pruebas funcionales

## 5.3 Pruebas de integración

Hoy en día no es suficiente con escribir pruebas unitarias para las aplicaciones Ruby On Rails, también es obligatorio realizar pruebas de integración que demuestren que la aplicación funciona bien en el escenario deseado y comprobar cómo interactúan entre sí varias partes de la aplicación.

Las pruebas de integración comprenden verificaciones asociadas a grupos de componentes, verificando que funcionan correctamente cuando son ensamblados para cumplir con una función concreta. Cada escenario se ha probado con el mayor número de entradas posibles y relevantes. Incluyéndose datos correctos e incorrectos para probar que el sistema responde adecuadamente a los errores. Según se han ido desarrollando las diferentes funcionalidades del proyecto se ha ido comprobando el flujo de trabajo, de manera integral, para comprobar que se obtiene un resultado coherente.

Ruby On Rails provee de un generador para crear test de integración. Estas pruebas son almacenadas en el directorio “./test/integration” de la aplicación, pero en este proyecto no se ha hecho uso de él.

## 5.4 Pruebas de usabilidad

Se define como la actividad de evaluar un producto o servicio probándolo con usuarios representativos. Normalmente, durante una prueba, los participantes tratarán de completar tareas típicas mientras los observadores observan, escuchan y toman notas. El objetivo es identificar cualquier problema de usabilidad, recopilar datos cualitativos y cuantitativos y determinar la satisfacción del participante con el producto.

Aunque el producto desarrollado sea de mucha calidad, si no es probada por un nutrido grupo de representantes de los usuarios, no se sabrá si es realmente útil para ese grupo de usuarios. O lo que es lo mismo, si un software es muy vanguardista, tecnológico o de calidad, no es apropiado si no tiene una utilidad. Para este proyecto se ha puesto especial énfasis en que la usabilidad de la web sea sencilla para cualquier tipo de usuario y para que además se visualice de forma correcta en la mayoría de los navegadores web del mercado actual.

Esto son algunos de los aspectos evaluados en las pruebas de usabilidad:

- Facilidad de aprendizaje, se ha buscado la interfaz de usuario más simple posible, con funcionalidades bien definidas, que sean fáciles de aprender y con consistencia. El diseño es sencillo de forma que el usuario sepa en todo momento donde está y las acciones que puede realizar.
- Flexibilidad, se ha intentado que el usuario escriba lo menos posible, pero hay situaciones, como la cadena de búsqueda, en la que es necesarios hacerlo.

- Tiempo de respuesta, una aplicación que no responda adecuadamente no es una aplicación válida, a este respecto se ha hecho todo lo posible para obtener una rápida respuesta del sistema.
- Reducción de la carga cognitiva, no se ha recargado con demasiado texto las páginas, excepto en la página de la evaluación, que al contener 31 preguntas es imposible quitarles el texto a las preguntas.
- Recuperabilidad, el sistema no muestra mensajes de error que no sean útiles para el usuario, o realiza indicaciones para subsanar algún error que hubiera cometido el usuario, como cuando va a dar un alta en alguna tabla y se olvida de introducir un valor en un campo que es obligatorio que lo tenga.

Todas las pruebas realizadas de manera informal han contado con un gran número de usuarios, principalmente familiares y amigos. Este grupo de usuarios cuentan con una gran diversidad de edades, formaciones educativas y formaciones tecnológicas. Gracias a estas pruebas se ha conseguido una valiosa información para ir haciendo modificaciones al sistema durante el desarrollo del proyecto, para hacerlo cada vez más intuitivo, completo y robusto para el usuario.

Se han realizado pruebas de funcionalidad sobre distintos sistemas operativos, navegadores y tamaños de pantallas en los dispositivos utilizados:

- Android
  - Tablet
  - Teléfono LG
- IOS
  - Tablet
  - iPhone 5/6 Plus/7
- Windows 10
  - Firefox
  - Explorer
  - Chrome
- Ubuntu
  - Firefox
  - Chrome
  - Opera
- MacOS
  - Safari
  - Firefox
  - Chrome

## 5.5 Pruebas de accesibilidad

Para las pruebas de accesibilidad existen infinidad de herramientas online gratuitas, se ha decidido utilizar algunas de las herramientas más extendidas para comprobar la accesibilidad de la aplicación web por personas que presentan alguna limitación, ya sea de índole física o técnica. Se ha asegurado la navegación de formularios por teclado y con el uso exclusivo del tabulador y los retornos de carro. También que las combinaciones de colores cuenten con el contraste suficiente y obviar el uso de colores rojos y azules para ayudar a los usuarios con daltonismo o problemas relacionados con el color.

La primera herramienta de validación que se ha utilizado es eXaminator, esta es una herramienta online que evalúa las pautas de accesibilidad en los contenidos HTML y CSS de la página, usa como referencia la WCAG 2.0 y clasifica el resultado final en una escala de 1 a 10. La página web ha obtenido una puntuación de 10 de 10 posibles, indicando que, de 10 pruebas realizadas, se obtienen 10 con un resultado excelente, 0 mal y 0 muy mal.

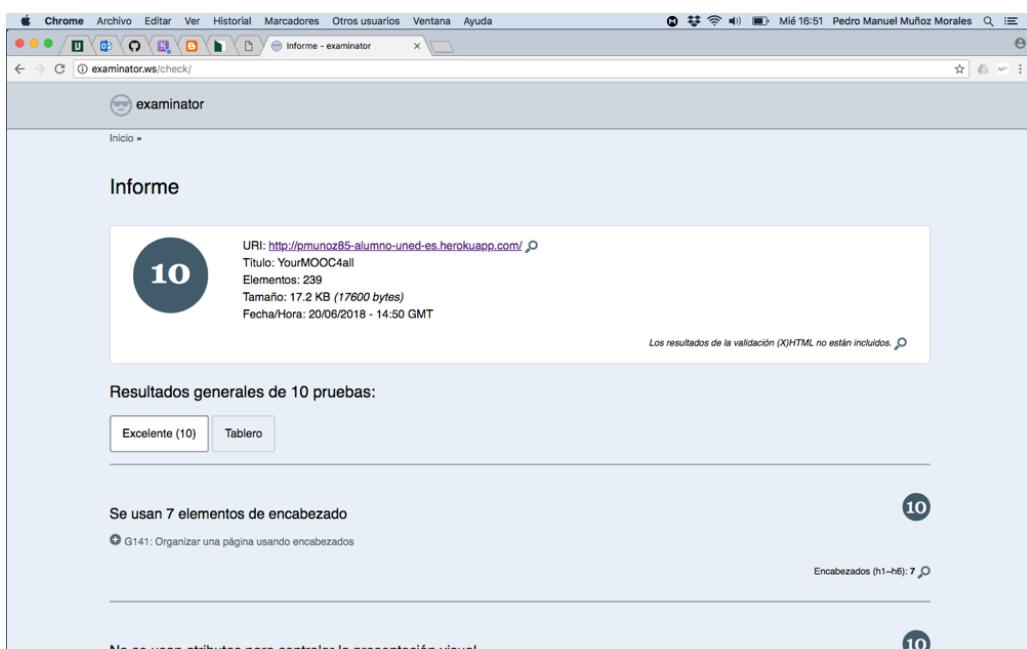


Figura 51 – eXaminator

Otra herramienta utilizada para validar la accesibilidad ha sido TAW, este es un programa de evaluación automática de los niveles WAI de accesibilidad desarrollado por la Fundación CTIC, que sirve de ayuda a los desarrolladores de páginas web para que puedan cumplir con las normas de accesibilidad. Con esta herramienta se puede analizar aquellos puntos referidos a las tres prioridades de la norma WCAG sobre accesibilidad. Revisa la WCAG 1.0 y 2.0. Se ha utilizado la versión online, pero también cuenta con un complemento para Mozilla Firefox.

Achecker es una herramienta de revisión de accesibilidad integral, que utiliza HTML para evaluar el contenido de una sola página. Se ha puesto a prueba la aplicación web insertando la

URL del despliegue en Heroku directamente en la herramienta online. Escanea de forma rápida el sitio web y genera un informe que identifica los posibles problemas de accesibilidad que pudiera tener. Esta desarrollada por ATRC (Adaptive Technology Resource Centre) y permite evaluar el contenido de la página conforme a las pautas de accesibilidad WCAG 1.0 y 2.0.

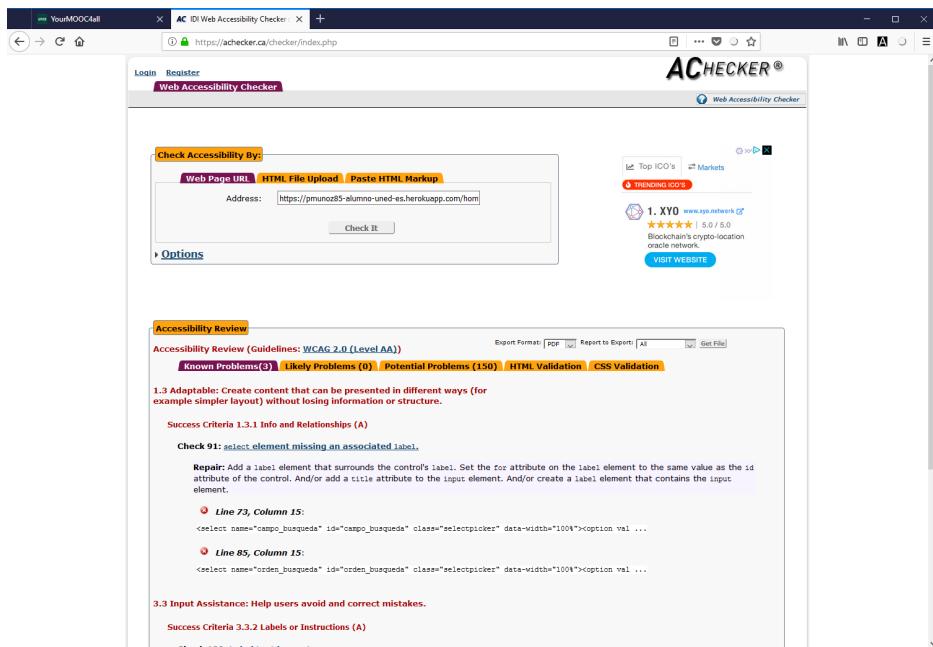


Figura 52 - Achecker

El W3C Markup Validation Service permite analizar documentos HTML y XHTML bien formados y válidos, esta validación es necesaria para asegurar la calidad técnica del sitio y verificar la compatibilidad con los navegadores. Esta herramienta ha encontrado un total de 7 errores y 346 advertencias en el CSS, que se deben reparar en versiones posteriores para obtener una mayor compatibilidad entre los diferentes navegadores del mercado.

## 5.6 Pruebas de seguridad

En este tipo de pruebas lo que se busca es recabar información sobre la confidencialidad, la integridad y la disponibilidad de los datos, buscando identificar amenazas y riesgos de un sistema. Estas pruebas consisten en revisar las aplicaciones en busca de vulnerabilidades.

Una vez que se han ejecutado las pruebas de seguridad es posible medir y cuantificar los riesgos a los cuales se ven expuestos la aplicación. Todas las aplicaciones tienen una exposición de seguridad, sin ir más lejos el propio usuario que la maneja, pero en concreto las aplicaciones web están expuesta de una manera muchísimo mayor, ya que su exposición es a nivel global. Para las pruebas se ha utilizado una herramienta llamada OWASP Zed Attack Proxy, la cual se basa en un proyecto llamado Open Web Application Security Project, el cual especifica los diez riesgos de seguridad más importantes de las aplicaciones web.

OWASP TOP 10 - 2017 RELEASE
A1 - Injection
A2 - Broken Authentication and Session Management
A3 - Cross-Site Scripting (XSS)
A4 - Broken Access Control
A5 - Security Misconfiguration
A6 - Sensitive Data Exposure
A7 - Insufficient Attack Protection
A8 - Cross-Site Request Forgery (CSRF)
A9 - Using Components with Known Vulnerabilities
A10 - Underprotected APIs

Figura 53 - OWASP TOP 10

Se ha realizado una prueba estándar en local con la herramienta ya que no se cuenta con los conocimientos suficientes para profundizar en los resultados obtenidos por dicha herramienta, tampoco es recomendable hacer uso de la herramienta cuando la aplicación está en producción, ya que el rendimiento y la estabilidad se verían comprometidas. Aquí se puede observar como la herramienta va navegando a través de los diferentes enlaces intentando descubrir ficheros ocultos usando diferentes técnicas como la lectura del fichero robots.txt, spider, etc.

The screenshot shows the OWASP Zed Attack Proxy (ZAP) interface. The main window displays a scan results summary. On the left, there's a tree view of the application structure and a list of detected vulnerabilities. One specific vulnerability is highlighted: 'Cross-Domain JavaScript Source File Inclusion' at 'http://192.168.120.27.3000/users/sign\_in'. The details pane on the right provides a detailed breakdown of this finding, including the URL, risk level (Low), confidence (Medium), and a snippet of the affected code. The code snippet shows a script tag pointing to a Google API endpoint.

Figura 54 - OWASP Zed Attack Proxy 1

Figura 55 - OWASP Zed Attack Proxy 2

## 5.7 Pruebas manuales

Para las pruebas manuales se han realizado de manera sistemática, una serie de test haciendo una exploración de grafos en anchura de todas las posibles acciones con las que cuentan los usuarios, tanto los no registrados, como los registrado y los administradores. Esta exploración en anchura se comienza siempre en la página de inicio del sitio web y se va explorando todas las posibles acciones que el usuario es capaz de realizar, esto se realiza de una forma secuencial y estructurada. En este tipo de pruebas se ha concentrado la mayor cantidad del tiempo de las pruebas, ya que las demás han sido pruebas muy superficiales, con un valor esencialmente académico. Este tipo de pruebas no es muy riguroso, pero es la manera más natural de probar cualquier producto fabricado. Tampoco tiene mucha fiabilidad ya que el grupo de personas que lo ha realizado es muy pequeño en comparación con la cantidad y variedad de usuarios que existe en Internet.

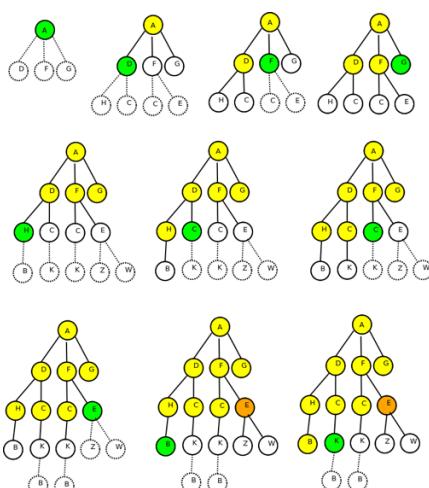


Figura 56 - exploración de grafo en anchura

## 5.8 Otras pruebas

Por último, se quiere destacar que existen otros tipos de pruebas que no son tan comunes como las mencionadas, pero son también muy recomendables, como son las pruebas de cobertura, las cuales informan de las pruebas que no han sido implementadas, para que se detecten las pruebas que son necesarias para añadir a tu conjunto de pruebas.

También se dispone de los test de mutación, que consiste en un software que realiza cambios en tu código fuente y lanza todos los test unitarios que se han implementado para esa parte del código. Si los test fallan eso significa que todo está correcto, pero, si los test unitarios no detectan ningún fallo, el test de mutación ha encontrado un fallo.

Para terminar con este apartado no se puede olvidar los Mock, que son objetos que imitan el comportamiento de objetos reales de forma controlada. Se usan para probar a otros objetos en test unitarios que esperan mensajes de una clase en particular para sus métodos. En este proyecto no se han utilizado ninguna de estas pruebas, por carecer de los conocimientos necesarios para manejar dichas herramientas e interpretar los resultados.

# 6 Planificación

Para la planificación del proyecto se ha seguido un modelo evolutivo incremental, este se divide en cuatro partes que son: el análisis, el diseño, la codificación y la fase de pruebas. Con esto se mantiene un contacto permanente con el cliente, en nuestro caso con el tutor, verificando cada poco tiempo que el proyecto cubre los requisitos impuestos por el proyecto.

El proyecto ha contado con una reunión semanal con el tutor, con lo que se ha conseguido ir haciendo pequeños avances durante todo el proceso sin perder la motivación ni sufrir atascos en el desarrollo del mismo. Este contacto permanente ha hecho que el proyecto cumpla con todos los requisitos iniciales a un ritmo muy alto y de manera constante.

## 6.1 Modelo incremental

Los modelos evolutivos son iterativos por naturaleza, permitiendo un desarrollo de versiones cada vez más completas y complejas, hasta llegar al producto final deseado.

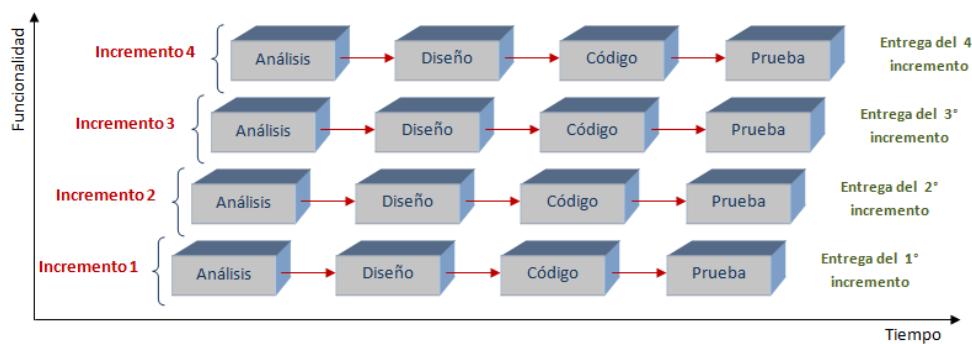


Figura 1: El Modelo Incremental

Figura 57 - modelo incremental<sup>27</sup>

Las características más importantes del modelo incremental son:

- Los requisitos iniciales deben estar bien definidos al principio.
- Se debe proporcionar al usuario un conjunto limitado de funcionalidades lo antes posible, expandiendo esa funcionalidad en entregas posteriores.
- Aplicar de forma iterativa el modelo en cascada, realizando iteraciones consecutivas sobre el propio modelo en casada.
- Cada iteración produce una nueva funcionalidad o refina una ya existente hasta que se consiga completar los requisitos iniciales.
- Los incrementos nuevos incrementos se cimientan en las iteraciones anteriores verificadas por el cliente.

<sup>27</sup>Modelo Incremental, Cristian Bermudez, Erika Garrido y Natalia Lara.  
Recuperado de <https://procesossoftware.wikispaces.com/Modelo+Incremental>

## 6.2 Planificación temporal

Todo proyecto requiere de una planificación en su ejecución, en este apartado se detalla la cronología de los hitos realizados y los tiempos estimados. Se ha empleado para ello una media semanal de 26 horas. El proyecto da comienzo el 1 de enero de 2018, finalizando en 25 semanas.

### Planificación

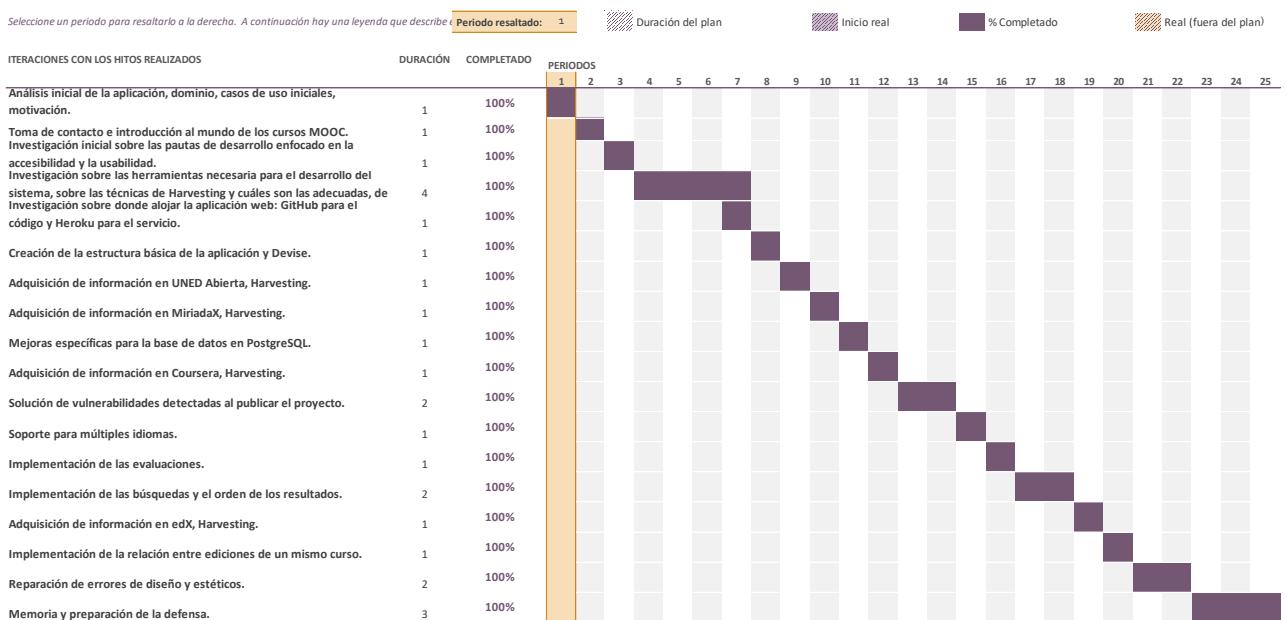


Tabla 17 - planificación temporal

Se ha utilizado el log de Git para la realización de los hitos, coincidiendo la finalización de un hito con la creación de un commit. Los comentarios aportados por en los commit no han sido muy acertados por lo que se han sustituido por otros más descriptivos. Estos periodos de tiempo sirven para acotar las iteraciones del desarrollo, conteniendo cada iteración: el análisis, el diseño, la codificación y la fase de pruebas. La mayoría de las iteraciones han sido de una semana de duración, coincidiendo con la reunión semanal que se ha tenido con el tutor.

Con el comando “rake stats” se puede saber la cantidad de líneas de código, LOCs, clases, métodos y demás, unido a la estimación en tiempo y valorando la complejidad, se podría valorar económicoamente este proyecto.

Name	Lines	LOC	Classes	Methods	M/C	LOC/M
Controllers	2293	1403	25	112	4	10
Helpers	103	81	0	12	0	4
Models	794	593	11	44	4	11
Mailers	0	0	0	0	0	0
Javascripts	6088	3974	0	515	0	5
Libraries	320	228	3	23	7	7
Controller tests	419	333	11	0	0	0
Helper tests	0	0	0	0	0	0
Model tests	70	30	10	0	0	0
Mailer tests	0	0	0	0	0	0
Integration tests	0	0	0	0	0	0
Total	10087	6642	60	706	11	7
Code LOC: 6279		Test LOC: 363	Code to Test Ratio: 1:0.1			

Figura 58 - LOCs



# 7 Conclusiones y trabajos futuros

En este apartado se detallan las conclusiones obtenidas después de haber realizado el proyecto y las líneas de trabajo que se podrían seguir aplicando al mismo y que no se han llevado a cabo por falta de tiempo.

## 7.1 Conclusiones

Como colofón a este proyecto se puede decir que la aplicación web fue concebida para compartir la experiencia de los usuarios, en relación con los cursos MOOC con los que hayan tenido algún tipo de relación. Al compartir estos conocimientos con la comunidad de internautas a través de las valoraciones, los nuevos alumnos en potencia de un curso MOOC en particular tendrán más información sobre a priori para tomar una correcta decisión de cara a la matriculación.

Por otro lado, al realizar una aplicación en la cual los alumnos de cursos MOOC pueden intercambiar experiencias involucra que, cuanto más se utilice la aplicación más cumplirá esta con su objetivo, al justar la puntuación del curso a la realidad. Se ha hecho un esfuerzo por tener una interfaz de usuario versátil, para que pueda ser accesible por la mayor cantidad de alumnos posible, que además resulte sencilla de utilizar con accesos rápidos a las funciones más comunes y esté soportado por la totalidad de navegadores actuales. En este proyecto se ha construido un sistema totalmente operativo y funcional, en el cual muchas personas con necesidades de formación serán capaces de localizar de una forma rápida los cursos MOOC en los que esté interesado. Además, se puede afirmar que la herramienta analizada, diseñada, implementada, testeada y puesta en producción, equivaldría a un proyecto real de una aplicación comercial al completo.

En cuanto al apartado de las tecnologías utilizadas se debe remarcar el uso de metodologías ágiles, porque el ciclo de vida se ha ido repitiendo en el tiempo hasta lograr que el proyecto estuviera completado. Este desarrollo se ha llevado a cabo con tecnologías de código abierto, fusionando tecnologías web como http, CSS, Response Design, JavaScript con tecnologías de programación como Ruby y Ruby On Rails también se ha utilizado conocimientos de bases de datos como SQL para MySQL y PostgreSQL. La extracción de información desde los diferentes sitios web oferentes de cursos como: MiriadaX, UNED Abierta, Coursera y edX, ha sido uno de los mayores factores de riesgo asumido en el proyecto, ya que no se conocía a priori si era posible llevarlo a cabo de una manera óptima.

La elección de Ruby On Rails como Framework de desarrollo ha sido un gran acierto, solo puedo destacar lo cómodo y sencillo que resulta el adaptarse a este Framework, el cual facilita y acelera enormemente la función del desarrollador mediante unas configuraciones de inicio muy efectivas, una estructura muy clara y sobre todo el uso de un lenguaje que resulta

intuitivo y funcional como es Ruby. Estas y otras muchas razones hacen posible que un desarrollador pueda empezar a trabajar forma productiva en Ruby On Rails muy pronto. También se debe destacar que cuenta con una comunidad muy activa en la resolución de problemas y que crece de forma continua.

## 7.2 Trabajos futuros

Como línea principal de trabajo que podría aplicarse a este proyecto sería, convertir esta aplicación web en una aplicación para telefonía móvil, tanto Android como IOS, pero esto significaría una cantidad ingente de horas en aprender dos lenguajes de programación diferentes y entornos de desarrollo también diferentes. Algo inviable para un solo programador.

Sin embargo, existe una tecnología, las Progressive Web Apps, que son el futuro de las aplicaciones web. Se basan en reducir cada vez más la línea que separa la web del mundo nativo. Una PWA utiliza las últimas tecnologías disponibles en los navegadores para ofrecer una experiencia en móviles lo más parecida a la de una aplicación nativa.



Figura 59 - Progressive Web<sup>28</sup> Apps 1

El usuario no será capaz de percibir si está utilizando una aplicación web u otra nativa, porque las PWA ocultan la barra de navegación, se muestran como un ícono más en el móvil y utilizan todo lo que el dispositivo móvil ofrece, como son: la cámara, las notificaciones push, el GPS, el acelerómetro, etc. También funcionan sin conexión a internet apoyándose en la caché para que el usuario no perciba la ausencia de conexión.

<sup>28</sup> Paweł Cała, Emil Waszkowski, What Are Progressive Web Apps?. Recuperado de <https://www.futuremind.com/blog/what-are-progressive-web-apps>

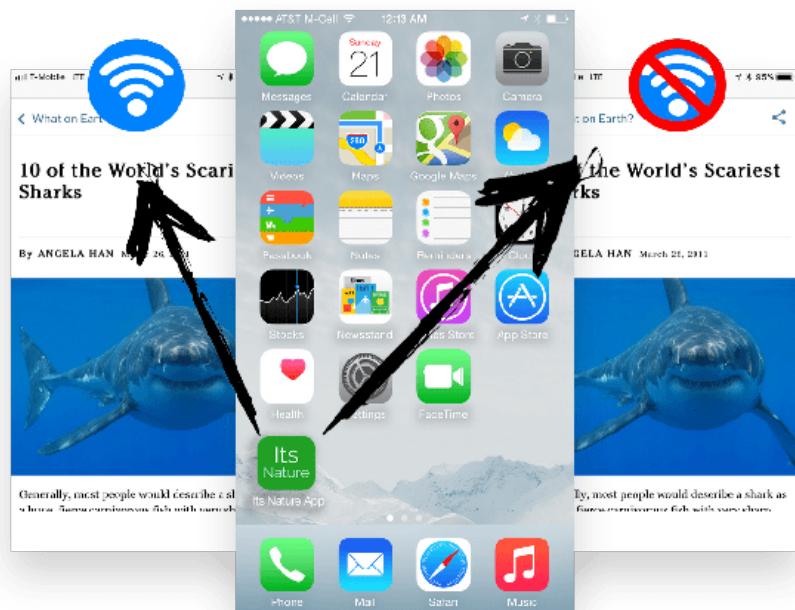


Figura 60 - Progressive Web<sup>29</sup> Apps 2

Para conseguirlo, las PWAs se basan en unos conceptos bastante simples: el Responsive Web Design, los Service Workers, la App Shell, el Manifiesto de aplicación y la Caché. Las páginas web de Twitter y Facebook son dos buenos ejemplos de aplicaciones PWA. En ellas puedes hacer prácticamente lo mismo que en las aplicaciones nativas para Android o iOS, pero sin necesidad de instalar nada: solo es necesario disponer de un navegador con conexión a Internet.



Figura 61 - Progressive Web<sup>30</sup> Apps 3

<sup>29</sup> EZOIC, Intelligent PWA Converter. Recuperado de <https://www.ezoic.com/es/intelligent-pwa-converter/>

<sup>30</sup> drptalk, Adam Hill – Front-End Developer at drpdigital. Google Academy: Progressive Web Apps (PWA). Recuperado de <https://drptalk.com/2017/11/29/google-academy-progressive-web-apps-pwa/>

Otras posibles ampliaciones para este proyecto serían:

1. La traducción a más idiomas para que así se pueda llegar a un mayor número de usuarios y aprovechar el alcance de Internet.
2. También tener la posibilidad de poder unirse a un grupo de chat con los alumnos que estén matriculados en un mismo curso. Esto sería algo así como una super aula para que los alumnos que se encuentran cursando un mismo MOOC.
3. Otra posible mejora para el sistema sería añadir exportaciones a csv, pdf o Excel de los cursos que se han encontrado en mis búsquedas en el sistema o los que tengo marcados como interesantes para mí. Esto me aportaría una visión en forma de tabla que para las comparaciones es más operativo.
4. Usar el correo electrónico para difundir información a nuestros usuarios registrados sobre cursos que acaban de ser añadidos en el sistema, o recomendaciones que el sistema le puede hacer según los cursos que son de su agrado.
5. Al no tener un modelo de negocio por el cual se pueda rentabilizar el sistema no es necesario incluir una pasarela de pago a través de algún banco o de PayPal, pero esta opción sería con objeto académico.

# 8 Manual

Este es un pequeño manual de uso para la aplicación web YourMOOC4all, en él se detallan todas las características que un usuario, de cualquier tipo, puede realizar.

## 8.1 Página de inicio



Figura 62 - manual - inicio

En esta página inicial se puede cambiar de idioma pulsando en el enlace correspondiente en la parte superior izquierda de la pantalla. En caso de que se tenga seleccionado el idioma español, solo se tendrá activo el idioma inglés.

En la parte central derecha de la pantalla existen tres enlaces con los que se puede ir a diferentes pantallas de la aplicación. Con el enlace “Identificarse” se va a la pantalla de identificación de usuarios, es obvio que solo se podrá identificar en el sistema un usuario que ya disponga de una cuenta en el sistema. Si se pulsa en el enlace “Registrarse” se va a la pantalla de registro del sistema. Por último, se puede observar el enlace “¿Olvidó su contraseña?” que permite al usuario recuperar la misma.

En la parte central izquierda se dispone de una etiqueta de texto para introducir un literal, el cual se buscarán las correspondencias en la base de datos de la aplicación, comparándolos con los campos que se indique en el desplegable inferior. Cabe destacar que cuando se quiera buscar un literal con varias palabras, pero no se quieran las coincidencias por separado sino unidas, por ejemplo ‘matemática cuántica’, se debe unir el literal con los símbolos “<” y “>”, así ‘<matemática cuántica>’.

En el desplegable con los campos de búsqueda se dispone de las siguientes opciones: Todos, Título del curso, Temática e Información.

Debajo de esta barra de búsqueda se encuentra un desplegable con el nombre del campo con el que se compara el literal para obtener los resultados solicitados. También se dispone de un desplegable para indicar el orden con el que se quiere mostrar el resultado de la búsqueda en el sistema. Se dispone de las siguientes opciones: Título del curso, Instituciones, Plataformas y Puntuación obtenida.

Más abajo se encuentran una serie de botones para ir a una página en concreto del resultado de la búsqueda, además de los botones Anterior y Siguiente. En el caso de que no se haya realizado ninguna búsqueda, el sistema muestra todos los cursos que tiene registrados. Cada curso se muestra dentro de un cuadro de color gris claro, dentro de este recuadro se muestra la imagen del curso, la votación del mismo si es que la tuviera, el título del curso, la institución que lo imparte, el tiempo de duración y la temática del mismo.

En la parte derecha del rectángulo de cada curso también se cuenta con un pequeño ícono con un pulgar hacia arriba, este ícono servirá para puntuar este curso en caso de que se esté registrado como usuario. El otro botón con una flechita inclinada hacia la derecha es un enlace a la página de inscripción del curso en la plataforma que lo imparte.

## 8.2 Resultado búsqueda

Figura 63 - manual – búsqueda 1

En esta página se puede apreciar que al introducir la palabra “contabilidad” y el orden para los datos obtenidos en “Puntuación obtenida”, se obtiene como resultado los cursos que han sido valorados con mayor puntuación en primer lugar.

## 8.3 Página de registro

The screenshot shows a web browser window with the URL [pmunoz85-alumno-uned-es.herokuapp.com](http://pmunoz85-alumno-uned-es.herokuapp.com). The page has a dark header with the text "YourMOOC4all" and "Diseño inclusivo de cursos MOOC y feedback útil". Below the header is a registration form titled "Registrarse". It contains three input fields: "Email", "Contraseña" (password), and "Confirmar contraseña" (confirm password). To the right of the form is a button labeled "Identificarse". At the bottom left is a note about the last update date (30-05-2018). Below the registration form are three navigation links: "Objetivos" (with a lock icon), "Proyecto" (with a house icon), and "Tecnologías" (with a chart icon). The "Proyecto" section includes the project lead's name (Pedro Manuel Muñoz Morales) and email (pmunoz85@alumno.uned.es).

Figura 64 - manual - registro

En esta página se encuentra un formulario para llenar con el correo electrónico y la contraseña que se va a utilizar como credenciales en la cuenta, esta contraseña tiene una restricción de seguridad y es que tiene que tener más de seis caracteres como mínimo, no importa si son números letras o símbolos.

Al pulsar en el botón “Registrarse” se pasa a formar parte del sistema quedando registrado en el mismo.

## 8.4 Página de identificación

The screenshot shows a web browser window with the URL [pmunoz85-alumno-uned-es.herokuapp.com](http://pmunoz85-alumno-uned-es.herokuapp.com). The page has a dark header with the text "YourMOOC4all" and "Diseño inclusivo de cursos MOOC y feedback útil". Below the header is a login form titled "Identificarse". It contains two input fields: "Email" and "Contraseña" (password), and a checkbox for "Recuérdame". To the right of the form is a button labeled "Registrarse" and a link "¿Olvidó su contraseña?". At the bottom left is a note about the last update date (30-05-2018). Below the login form are three navigation links: "Objetivos" (with a lock icon), "Proyecto" (with a house icon), and "Tecnologías" (with a chart icon). The "Proyecto" section includes the project lead's name (Pedro Manuel Muñoz Morales) and email (pmunoz85@alumno.uned.es), and the director's name (Francisco Iniesta Carrasco) and affiliation (The Open University, Reino Unido).

Figura 65 - manual - identificación

Aquí es donde una vez registrado el usuario en visitas posteriores al registro deberá identificarse aportando las credenciales introducidas en la página de registro, como ya se ha visto en el apartado anterior. Una vez que se ha introducido el correo electrónico y la contraseña, se puede pulsar el botón “Identificarse” para entrar en el sistema como usuario registrado.

## 8.5 Recuperación de contraseña



Figura 66 - manual – recuperar contraseña 1

Si se ha olvidado la contraseña, pero se recuerda el correo electrónico con el que se realizó el registro, en esta página se puede pedir al sistema que se envíe un correo electrónico con un enlace único y con caducidad en el tiempo para que, si realmente es el propietario de ese correo electrónico y ha solicitado el cambio de contraseña pueda hacerlo.

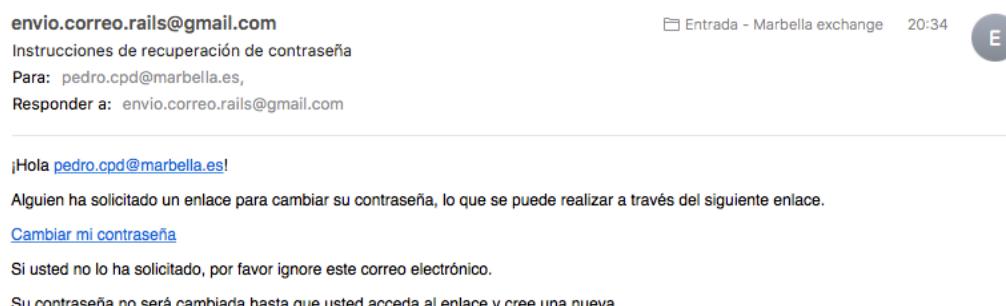


Figura 67 - manual – recuperar contraseña 2

Una vez que se rellena la etiqueta de texto con el correo electrónico y se pulsa sobre el botón “Envíame las instrucciones para resetear la contraseña”. El sistema enviará el correo, después el usuario debe consultar su correo electrónico y buscar el correo que le ha enviado el

sistema. En este correo existe un enlace que se debe pulsar, en caso de hacer clic en el enlace se abrirá una pestaña en el navegador donde se puede cambiar la contraseña del sistema.

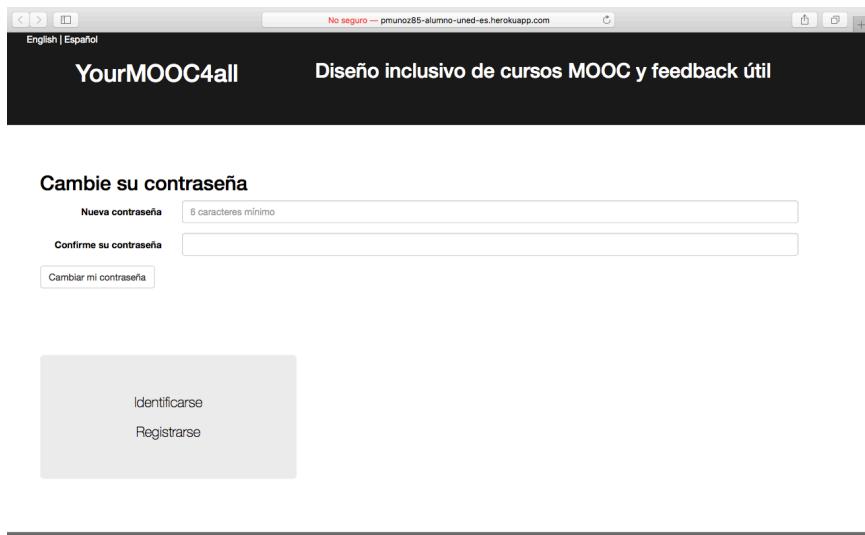


Figura 68 - manual – recuperar contraseña 3

## 8.6 Ver información sobre un curso

A screenshot of a web browser window showing course details. The course title is 'Contabilidad para no contadores' with a rating of 'Calificación: 4.5 ★★★★☆'. It's offered by 'Plataforma: Coursera' and 'Institución: Universidad Nacional Autónoma de México'. The course description is in Spanish, mentioning it's about accounting principles and financial statements. It includes sections for 'Temática', 'Información', and 'WEEK 1'. The URL is https://www.coursera.org.

Figura 69 – manual – información del curso 1

Objetivos aprendizaje: Beginner  
Conocimientos previos:  
Destinatarios y nivel: Who is this class for: Este curso es para todas las personas que deseen desarrollar conocimientos sobre los aspectos básicos de la contabilidad. Estos, pueden ser aplicables para operar un negocio particular, una pequeña empresa o la vida contable cotidiana.  
Actividades evaluación: Pass all graded assignments to complete the course.  
Esfuerzo estimado: 5 semanas de estudio, 1 a 3 horas por semana  
Próxima edición: Starts Jun 04  
Lenguaje de signos: No  
Precio auditado:  
Precio credencial:  
Edición anterior:  
Enlace al curso: [C](#)

[Atrás](#)

**Transcripciones**  
No disponible

**Audios**  
No disponible

**Subtítulos**  
No disponible

**Evaluaciones**

Motivación      Representación      Expresión      Texto libre

Figura 70 - manual - información del curso 2

Si se pulsa sobre el título de cualquier curso, se ve toda la información que el sistema tiene registrada sobre este curso, por ejemplo: idiomas en los que está el audio, los subtítulos, las transcripciones y las evaluaciones que han hecho los usuarios registrados sobre el curso.

## 8.7 Inicio para usuario registrados e identificados

Utilice el buscador para encontrar los cursos en los que está interesado

Utiliza < y >, para unir palabras que deban ir juntas

Selección del campo de búsqueda: Todos

Orden para los datos obtenidos: Puntuación obtenida

Cerrar sesión

pedro.cpd@marbella.es

« Anterior 1 2 3 4 5 6 7 8 9 ... 160 161 Siguiente »

Curso sin puntuación, sé el primero en valorarlo

Learn Spanish: Basic Spanish Vocabulary Specialization

Figura 71 – manual – inicio de usuario identificado

Si ya se está identificado en el sistema, se cuenta con cuatro opciones nuevas, de las cuales la primera es obvia, se puede cerrar la sesión de usuario registrado, esto llevaría al usuario a la pantalla de inicio comentada al principio de este apartado. La segunda es que se puede editar la información que existe sobre el usuario en el sistema. La tercera es que se puede evaluar un curso y aportar la opinión que un usuario tiene sobre él en forma de texto libre. Y, por último, la cuarta que es poder marcar un curso como interesante para el usuario,

esto significa que cada vez que el usuario vuelva al sistema este le recuerda los cursos marcados con anterioridad para tenerlos siempre visibles.

## 8.8 Editar información de usuario

Editar usuario

Email: pedro.cpd@marbella.es

Contraseña: Déjala en blanco si no quiere cambiarla

Confirmar su contraseña:

Contraseña actual: Necesitamos su contraseña actual para confirmar los cambios

Actualizar

Cancelar mi cuenta

¿No está contento con nuestro servicio?

Cancelar mi cuenta

Volver

Fecha de la última edición 30-05-2018

Objetivos Proyecto Tecnologías

Figura 72 - manual – información de usuario

En este formulario se puede cambiar el correo electrónico o la contraseña del usuario, solo si antes se introduce la contraseña actual en el sistema. También se puede dar de baja en el sistema si el usuario no está muy contento con el mismo.

## 8.9 Evaluar curso

Valora tu motivación a lo largo del MOOC

1 2 3 4 5

1. ¿Puedes participar cuando quieras en las discusiones o actividades y trabajar con total libertad de tiempo? +

2. ¿Las actividades propuestas concuerdan con lo que deseabas aprender, dándote la posibilidad de explorar el contenido y ser creativo? +

3. ¿La información sobre las actividades se notifica con anterioridad (al comienzo del MOOC o con emails), hay acceso a un calendario con toda la información? +

4. ¿Al principio del MOOC tienes espacio para poder formular lo que esperas aprender? +

5. ¿Sí diferencia el nivel de dificultad en las actividades propuestas en el MOOC? +

6. ¿Puedes discutir lo que deseas aprender en el MOOC con otros compañeros? +

7. ¿Las respuestas por parte de los facilitadores y compañeros es positiva y orientada a ayudarte? +

8. ¿Los test proporcionan una reorientación que ayuda tu aprendizaje? +

9. ¿Se facilitan lugares para liberar estrés y poder hablar libremente sobre las dificultades encontradas? +

10. ¿Existe algún tipo de ayuda en caso de que no hayas podido participar en todo el MOOC? +

Incluye tus valoraciones relacionadas con la motivación en el MOOC

Valora la representación del contenido a lo largo del MOOC

Figura 73 - manual – evaluar curso

En esta página se aporta la puntuación y la valoración respecto al curso que el usuario ha realizado o está realizando. Esta puntuación pasará a formar parte de la puntuación media y

los comentarios valorando el curso se podrán ver en las búsquedas donde aparezca este curso y en la información detallada del mismo.

## 8.10 Página de inicio para el administrador

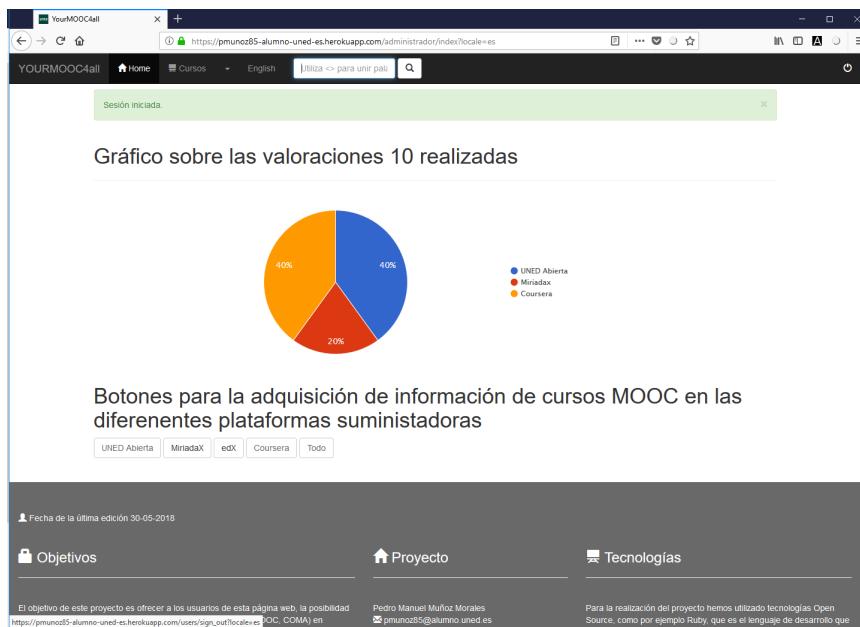


Figura 74 - manual – inicio administrador

Está página es el home del administrador, es la página que se muestra cuando un administrador se identifica en el sistema. En ella se pueden encontrar unos botones para comenzar con la adquisición de los cursos de las diferentes plataformas de manera asíncrona y otro con la palabra “Todo”, que busca en todas las plataformas registradas en el sistema haciendo un recorrido por todas ellas.

En el centro de la página se cuenta con información en forma de gráfico para las votaciones realizadas, agrupados por las diferentes plataformas a las que pertenecen los cursos evaluados. En la parte superior se puede ver un menú con las diferentes opciones disponibles, como son: El home, los cursos, las instituciones, las plataformas, los idiomas, los usuarios y un enlace a la edición de la información personal del propio administrador.

En la barra superior también se observa una etiqueta de texto en la cual se puede introducir un literal para hacer la búsqueda pulsando sobre el botón con una lupa. El resultado obtenido en dicha búsqueda será mostrado en la parte inferior de la página inicial, después de los botones de adquisición y antes del pie de la página donde se tiene la información relacionada con nuestro proyecto.

## 8.11 Gestión de cursos

The screenshot shows a web-based course management system. At the top, there's a header with the title 'YOURMOOC4all' and a navigation bar with links for 'Home', 'Cursos', 'English', and a search bar. Below the header is a pagination bar with links from 'Anterior' to 'Siguiente' and page numbers 1 through 33. A blue button labeled '+ Nuevo curso' is located in the top right corner. The main content area displays a table of courses with the following columns: Nombre (Name), Plataforma (Platform), Imagen (Image), Institución (Institution), Temática (Topic), Estudio estimado (Estimated Study), and Enlace (Link). There are five courses listed:

Nombre	Plataforma	Imagen	Institución	Temática	Estudio estimado	Enlace
Fundación MAPFRE: Actualizaciones para la enseñanza de la seguridad vial y la prevención de accidentes	UNED Abierta		Fundación MAPFRE	IMPORANTE: Una vez se inicie el curso aparecerá un aviso donde deberás aceptar, siquieras, las condiciones de protección de datos.	25 horas	<a href="#">Enlace</a>
Curso Cero en Metodología de las Ciencias del Comportamiento	UNED Abierta		UNED	Se trata de recordar conceptos básicos para facilitar el estudio de las materias impartidas en los departamentos de Metodología de las facultades de Psicología. Realizaremos un repaso de ciertas habilidades matemáticas básicas importantes para afrontar un curso de Estadística.	25 horas	<a href="#">Enlace</a>
Autogestión en la escritura académica 2. Creatividad versus científicidad	UNED Abierta		UNED	Los contenidos son los siguientes:	25 horas	<a href="#">Enlace</a>
Claves contemporáneas de China (2º ed)	UNED Abierta		UNED	Este curso explica la realidad de la República Popular China desde su pasado hasta la actualidad. Los materiales del curso proporcionan al alumno los conocimientos básicos necesarios para comprender la realidad de RPCh y el papel que desempeña a nivel internacional.	25 horas	<a href="#">Enlace</a>
Los orígenes históricos y culturales de la Unión Europea	UNED Abierta		UNED	SABIAS QUILÉ estas palabras aparecen en una entrevista realizada en El mundo (15 de junio, 2013) a Paolo Savona, Presidente del Fondo Interbancario de Tutela de los Depósitos de la Unión Europea	25 horas	<a href="#">Enlace</a>

Figura 75 - manual - gestión de cursos

Desde esta página se puede administrar todo lo relacionado con los cursos. En principio se ve una lista paginada con un máximo de 20 cursos por página, con todos los cursos que el sistema tiene registrados. Si se hace clic sobre su nombre el sistema muestra toda la información detallada del curso. También se puede crear un curso desde cero con el botón “Nuevo curso”.

## 8.12 Editar curso

The screenshot shows a detailed edit form for a course. At the top, there's a header with the title 'YOURMOOC4all' and a navigation bar with links for 'Home', 'Cursos', 'English', and a search bar. The main content area has several input fields and dropdown menus:

- Oculto: A checkbox labeled 'No se muestra en las búsquedas de los usuarios, si en la del administrador del sitio.'
- Identificador: Input field containing '/courses/course-v1:UNED+CCasComport\_01+2017/about'
- Url: Input field containing 'https://edra.uned.es'
- Nombre: Input field containing 'Curso Cero en Metodología de las Ciencias del Comportamiento'
- Imagen url: Input field containing 'https://edra.uned.es/asset-v1:UNED+CCasComport\_01+2017+type@asset+block@portada\_cc\_comportamiento.jpg'
- Plataforma: Dropdown menu set to 'UNED Abierta'
- Institución: Dropdown menu set to 'UNED'
- Temática: Text area containing 'Se trata de recordar conceptos básicos para facilitar el estudio de las materias impartidas en los departamentos de Metodología de las facultades de Psicología. Realizaremos un repaso de ciertas habilidades matemáticas básicas importantes para afrontar un curso de Estadística.'
- Información: Text area containing 'AVISO: Este curso es de autoaprendizaje por lo que no habrá apoyo docente. Cualquier duda que se tenga sobre los contenidos deberá formularse en los Foros y debatirse entre el conjunto de los compañeros. El equipo docente sólo resolverá los errores que pudieran encontrarse.'
- EQUIPO DOCENTE: Text area containing 'Todos los profesores del equipo docente pertenecen al Departamento de Metodología de las Ciencias del Comportamiento.  
Mª Araceli Maciá  
José Manuel Reales'
- Objetivos aprendizaje: Large text area for learning objectives.
- Conocimientos previos: Large text area for previous knowledge.

Figura 76 - manual – editar curso 2

En esta página se puede modificar los datos relacionados con un curso, desde la imagen que pertenece a ese curso o si se quiere se puede ocultar este curso para las búsquedas de los usuarios. Se recuerda que está opción oculta el curso para las búsquedas de los usuarios, pero nunca para los administradores. Una vez realizados los cambios se puede guardar o cancelar con los botones de la parte inferior.

Figura 77 - manual – editar curso 2

Uno de los campos editables en los cursos es el que contiene información sobre las ediciones anteriores. Cuando se conoce que un curso adquirido recientemente ya cuenta con una edición anterior, se debe utilizar el desplegable “Edición anterior” para indicar cuál es el curso al que pertenece esta nueva edición.

Por ejemplo, si existe el curso “Curso cero de Fundamentos de Biología para Psicólogos”, el curso “Curso cero de Fundamentos de Biología para Psicólogos 2<sup>a</sup> edición” y el cursor “Curso cero de Fundamentos de Biología para Psicólogos 3<sup>º</sup> edición”, y se descubre una cuarta edición de este curso, se debe seleccionar en el desplegable la tercera edición de este curso, no la primera ni la segunda sino la tercera. Esto es muy importante porque la conexión entre un curso con diferentes ediciones se realiza a modo de lista enlazada, uniéndose unos a otros.

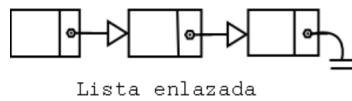
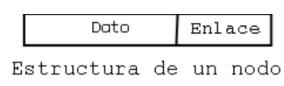


Figura 78 - manual – lista enlazada

## 8.13 Añadir/modificar/eliminar idiomas

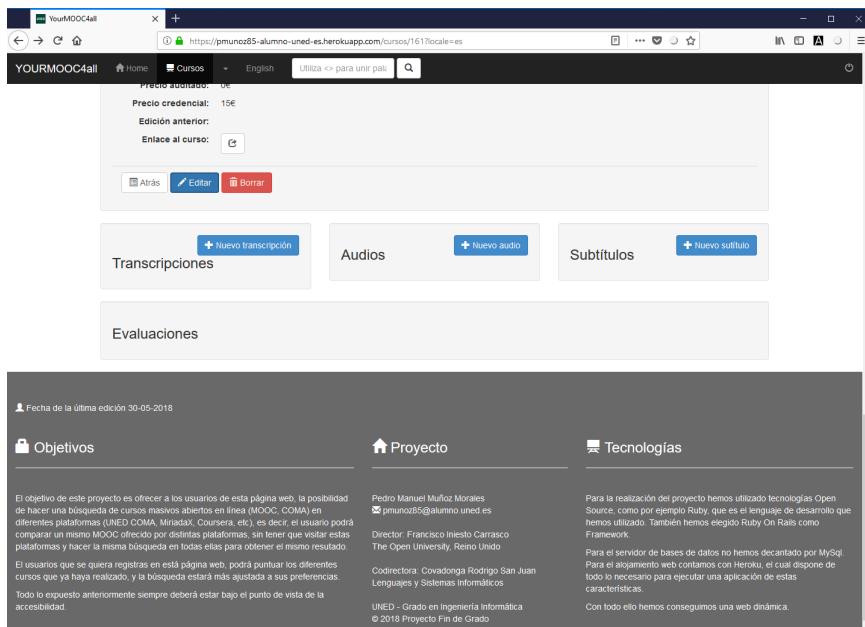


Figura 79 - manual - idiomas

En esta pantalla se observan unos botones en la parte inferior donde se puede añadir un idioma nuevo para las transcripciones, los audios o los subtítulos. Esta actividad solo la puede realizar un administrador ya que los algoritmos de adquisición de información automatizados no son capaces de realizar esta acción de manera desatendida.

Lo que se acaba de introducir referente a las transcripciones de los cursos, se debe aplicar a las demás tareas del administrador como son la gestión de las instituciones, las plataformas, los idiomas y los usuarios. Como la gestión de estas tablas es muy parecida a la de cursos se da por explicado dando por concluido el manual del usuario.



# REFERENCIAS BIBLIOGRÁFICAS

1. Iniesto, F.; Rodrigo, C. Strategies for improving the level of accessibility in the design of MOOC-based learning services. In: International Symposium on Computers in Education (SIEE), Salamanca, Spain. (2016)
2. Iniesto, F.; Rodrigo, C. Can user recommendations be useful for improving MOOCs accessibility? A project for inclusive design and profitable feedback. In: Open Education Global Conference 2016, Krakow, Poland. (2016)
3. Yuan, L.; Powell, S.: MOOCs and disruptive innovation: Implications for Higher Education. *eLearning Papers, In-depth*, 33, 2, 1-7 (2013)
4. Hill, P. Online Educational Delivery Models: A Descriptive View. *Edu-cause Review*, No 47(6), Pages 84-86. Retrieved from <http://www.educause.edu/ir/library/pdf/ERM1263.pdf> (2012)
5. Yang, D.; Sinha, T.; Adamson, D.; Rose, C. P.: Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In Proceedings of the 2013 NIPS Data-Driven Education Workshop (Vol. 10, p. 13) (2013).
6. Liyanagunawardena, T. R.; Parslow, P.; Williams, S.: Dropout: MOOC participants' perspective. *Proceedings of the European MOOC Stakeholder Summit 2014* (2014): 95-100.
6. de Waard, I., Gallagher, M. S., Zelezny-Green, R., Czerniewicz, L., Downes, S., Kukulska-Hulme, A., & Willems, J.: Challenges for conceptualising EU MOOC for vulnerable learner groups. *Proceedings of the European MOOC Stakeholder Summit 2014* (2014): 33-42.
7. CAST. Universal Design for Learning Guidelines version 2.0. Wakefield, MA (2011)
8. Lucas Carlson, Leonard Richardson: *Curso de Ruby, Ruby Cookbook* (2009)
9. Obie Fernandez, Kevin Faustino and Vitaly Kushner: *The Rails 4 Way* (2014)
10. Bruce A. Tate, Curt Hibbs: *Ruby on Rails* (2009)
11. Santiago Ponce Moreno: *Ruby on Rails. Desarrollo práctico de aplicaciones web* (2013)
12. Sam Ruby, Dave Thomas, David Heinemeier Hansson; Traducción de Teresa Samaniego Cabañas: *Desarrollo Web con Rails* (2009)
13. José López Quijado: *Domine HTML y DHTML* (2007)
14. Juan Diego Gauchat: *El gran libro de HTML5, CSS3 y JavaScript* (2012)