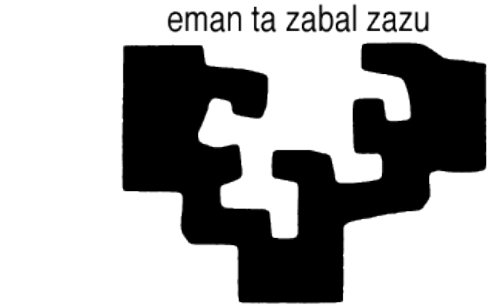


Facultad de Informática / Informatika Fakultatea

Aplicación web para el desarrollo de tutoriales paso a paso

Alumno: Lander Laparra Montecatine
Director: Jon Iturrioz Sánchez

Proyecto Fin de Carrera, julio 2014

RESUMEN

El proyecto fin de carrera de Ingeniería Informática tratará de la realización de una aplicación web destinada a la creación y visualización de tutoriales paso a paso de cualquier temática. Para la realización de este proyecto se utilizarán las últimas técnicas de desarrollo de aplicaciones web que sean de uso libre y que no conlleven costo alguno.

La elección de una aplicación web como proyecto de fin de carrera se fundamenta en la experiencia laboral adquirida en este campo desarrollando diferentes proyectos en Java EE para departamentos y entidades del Gobierno Vasco desde 2007.

Aunque tenía la posibilidad de elegir un proyecto interno en la empresa en la que desarrollo mi actividad laboral, me he decantado por un desarrollo autónomo. Las motivaciones de dicha elección vienen dadas por el reto que éste representa y por poder crecer profesionalmente tanto con los nuevos conocimientos que desarrolle en su implementación como con la obtención de la titulación de Ingeniería Informática.

Durante el proyecto se llevará a cabo un estudio previo de los Frameworks más punteros del momento y de los soportes necesarios para servir y desarrollar la aplicación de forma que su desarrollo y prueba sea lo más sencillo posible.

El siguiente documento aúna todos los aspectos necesarios para la realización de la memoria final del proyecto. Se incluyen en él: la planificación, desarrollo técnico, gestión del proyecto, bibliografía, etc.

Prueba la aplicación:

http://plapiz.noip.me/a83War

ÍNDICE DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. ANÁLISIS DE VIABILIDAD

1.1. Justificación del proyecto

1.1.1. Título

Oficialmente el título de este proyecto es "Aplicación web para el desarrollo de tutoriales paso a paso". De cara al usuario final (dado que se trata de una aplicación abierta al público) se necesita un nombre por el cual sea fácilmente identificado.

Para este fin se busca información sobre "*naming*", tan de moda en los tiempos que corren. El término *naming* se refiere a las técnicas para la creación del nombre de una marca. *Naming* (nombrar, poner nombre a una marca) sigue un proceso de creación de identidad de la marca, para que el producto se diferencie del resto.

Lo ideal es encontrar un nombre único, que no exista. Eso no garantiza que permanezca el nombre en exclusividad para la aplicación diseñada, pero de entrada dará una ventaja competitiva.

Un buen nombre debe insinuar el concepto, la esencia, el uso… de la aplicación. De este modo será más fácil saber de qué trata la marca en cuestión y qué tipo de servicios ofrece. No conviene que sea 100% literal/descriptivo ya que de ser así no lo podremos registrar (si los resultados pueden predecir un futuro comercial). Además, así facilitamos que el nombre de la aplicación se convierta en un nombre genérico.

El nombre ideal debe ser pegadizo, fácil de pronunciar y fácil de recordar para el público objetivo al que va dirigido.

Además, cuanto más simple y más corto sea el nombre, mejor. Lo ideal sería que fuera solamente una palabra (más fácil de recordar), pero si es altamente recomendable que no sean de más de tres.

No hay que olvidar que si la aplicación va a estar accesible desde diferentes países necesita un nombre internacional. Habrá que tener en cuenta también las diferencias dialécticas para que el nombre elegido no tenga connotaciones negativas en otros países ya que esto puede lanzar la aplicación al fracaso.

En base a todas estas técnicas de *namig* descritas, la elección del nombre se ha decantado por **Tipi-Tapa**, inspirándome en gran parte en el nombre de Coca-Cola.

1.1.2. Descripción general

Tipi-Tapa es una aplicación orientada a la creación y ejecución de tutoriales paso a paso online.

¿Qué es un tutorial? Los cursillos o tutoriales son sistemas instructivos de auto-aprendizaje que pretenden emular al maestro y muestran al usuario el desarrollo de algún procedimiento o los pasos para realizar determinada actividad.

1.1.3. Necesidades del negocio que satisface el proyecto

A título personal, en innumerables ocasiones me he visto en la tesitura de tener que hacer algo para lo cual no tengo conocimiento alguno, ya sea la elaboración de una receta de cocina, un cóctel, tapizar una silla o modificar la instalación eléctrica.

Todo lo que encontraba en Internet en su mayoría se ubicaba en blogs o foros, explicado con poco detalle o estaba destinado a entendidos en la materia puesto que gran parte de esos sitios son dedicados a temáticas concretas.

Si bien es cierto que existen muchas páginas de recetas dedicadas a tal uso con una buena gestión de ingredientes y otras variables, me vino a la mente: ¿Por qué no habrá ningún sitio dedicado a cualquier temática? ¿Acaso no puede cualquier tutorial descomponerse en un conjunto de pasos básicos para los cuales se necesiten un conjunto de materiales?

Así es como nació la idea original de este proyecto, no como una plataforma de e-learning dedicada al aprendizaje y a la evaluación de habilidades aprendidas sino como un índice de cualquier tipo de tutorial que esté bien esquematizado e indexado. Al fin y al cabo la necesidad de conocimiento de un individuo se puede extrapolar a la de varios.

1.2. Objetivos del proyecto

1.2.1. Objetivos

Creación de una aplicación web (cliente-servidor) así como todo el entorno de soporte de la misma en lenguaje Java con el uso de las últimas tecnologías de desarrollo.

El objetivo de esta aplicación será la creación y visualización de tutoriales on-line paso a paso y estará dividida de forma lógica en dos partes bien diferenciadas:

- Intranet para la administración de usuarios y contenido.
- Extranet para la visualización de contenidos publicados.

1.2.2. Restricciones

- La aplicación deberá tener un diseño sensible o responsivo que se adapte a cualquier resolución y que pueda ser visualizada correctamente en dispositivos móviles actuales (smartphones, tablets, etc.).
- El diseño de la interfaz debe ser comercial, fácil e intuitivo.

- El diseño de la interfaz debe ser sensible a eventos táctiles puesto que podrá manejarse en dispositivos con pantallas enfocadas a tal efecto.
- La gestión de permisos y la seguridad ha de ser robusta dado que un usuario sin permiso de acceso a la Intranet no deberá nunca poder entrar o realizar funciones de la misma.
- No tiene que servir páginas web pesadas para que sea lo más rápida posible.

1.3. Herramientas a usar en su elaboración

- **LibreOffice**

  LibreOffice es una suite ofimática libre y de código abierto desarrollada por The Document Foundation. Cuenta entre otras aplicaciones con un procesador de texto (Writer), un editor de hojas de cálculo (Calc) y un gestor de presentaciones (Impress) que se emplearán en el soporte documental del proyecto.

- **GIMP**

  GIMP (GNU Image Manipulation Program) es un programa de edición de imágenes digitales en forma de mapa de bits, tanto dibujos como fotografías. Es un programa libre y gratuito. Forma parte del proyecto GNU y está disponible bajo la Licencia pública general de GNU y GNU Lesser General Public License. Se empleará tanto para el soporte documental del proyecto como para los gráficos que se necesiten en el mismo.

- **StarUML**

  StarUML es una herramienta UML de código abierto que soporta la mayoría de los diagramas especificados en la notación UML 2.0. Se empleará para el soporte documental del proyecto.

- **Planner**

  Planner es una herramienta de administración de proyectos de propósito general que proporciona una gran variedad de funcionalidades, que están disponibles a través de cuatro pantallas distintas, llamadas vistas. Como se verá, se accede a las vistas haciendo clic en los iconos de la barra de herramientas de la izquierda de la ventana de Planner.

- **Apache**

  El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual.

  Apache es usado principalmente para enviar páginas web estáticas y dinámicas en la World Wide Web.

Instalado en el servidor, se empleará como ambiente de implantación en entorno de producción (admite conexiones remotas desde Internet).

- **JBOSS**

  JBOSS es un servidor de aplicaciones Java EE de código abierto implementado en Java puro. Al estar basado en Java, JBOSS puede ser utilizado en cualquier sistema operativo para el que esté disponible la máquina virtual de Java.

  JBOSS AS es el primer servidor de aplicaciones preparado para la producción y certificado J2EE 1.4, disponible en el mercado. Con una licencia GNU de código abierto, JBOSS AS puede ser descargado, utilizado, incrustado y distribuido sin restricciones por la licencia.

  Entre sus características, las que destacan para su elección son las siguientes:

  - Caché: utiliza cacheado en la capa de visualización para mejorar el rendimiento.
  - Compatibilidad 100% con J2EE 1.4 al utilizar JBoss AS.
  - Soporte de Frameworks existentes: los Portlets pueden utilizar Struts, Spring MVC, Sun JSF-RI, AJAX o MyFaces.
  - Fácil integración con Apache

  Instalado en el servidor y en la máquina de desarrollo, será el encargado de servir la aplicación web tanto en el entorno de producción como en el de desarrollo.

- **SSH Server**

  Instalado en el servidor permitirá la comunicación entre ambas máquinas (producción-desarrollo) mediante el protocolo SSH.

  SSH (*Secure SHell*, en español: intérprete de órdenes segura) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos y además también permite copiar datos de forma segura (tanto archivos sueltos como simular sesiones FTP cifradas).

- **Subversion Server**

  Subversion (SVN) es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo una licencia de tipo Apache/BSD.

  SVN permite al usuario crear, copiar y borrar carpetas con la misma flexibilidad con la que lo haría si estuviese en su disco duro local. Dada su flexibilidad, es necesaria la aplicación de buenas prácticas para llevar a cabo una correcta gestión de las versiones del software generado.

  Subversion puede acceder al repositorio a través de redes, lo que le permite ser

usado por personas que se encuentran en distintas computadoras. La posibilidad de que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivas ubicaciones fomenta la colaboración. Se puede progresar más rápidamente sin un único conducto por el cual deban pasar todas las modificaciones. Y puesto que el trabajo se encuentra bajo el control de versiones, no hay razón para temer que la calidad del mismo vaya a verse afectada. Si se ha hecho un cambio incorrecto a los datos, se puede deshacer ese cambio.

Instalado en el servidor será una utilidad de gran valor puesto que aparte de servir como copia de respaldo del código en desarrollo guarda un control de versiones. Así es posible deshacer modificaciones incorrectas o guardar fotos de momentos importantes en el desarrollo. Además, al estar alojado en el servidor, se podrá compartir toda la documentación necesaria con el tutor del proyecto pudiendo éste llevar un control libre sobre el estado del proyecto y la documentación.

- **Plataforma de desarrollo Eclipse**

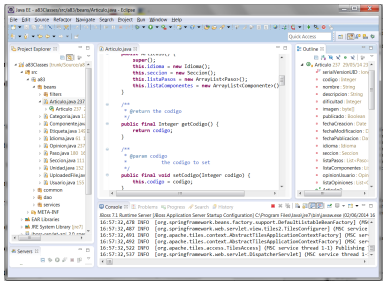


Ilustración 1: Eclipse IDE

Eclipse es un Entorno Integrado de Desarrollo, del inglés *Integrated Development Environment* (IDE), para todo tipo de aplicaciones, inicialmente desarrollado por IBM, y actualmente gestionado por la Fundación Eclipse. Junto con Netbeans es el IDE más utilizado para la creación de aplicaciones en Java.

Es una herramienta para el programador destinada principalmente para el desarrollo de aplicaciones Java. Es posible añadir nuevas funcionalidades a través de plugins como el Subclipse. Subclipse sirve para subir proyectos a repositorios y trabajar remotamente con otros usuarios controlando las versiones.

1.4. Fases, estimación de tiempos y estructura de descomposición de trabajo

1.4.1. Estructura de Descomposición de Trabajo (EDT)

El propósito de una EDT es organizar y definir el alcance total aprobado del proyecto según lo declarado en la documentación vigente. Su forma jerárquica permite una fácil identificación de los elementos finales, llamados "Paquetes de Trabajo". Se trata de un elemento exhaustivo en cuanto al alcance del proyecto. La EDT sirve como la base para

la planificación del proyecto. Todo trabajo del proyecto debe poder identificarse en una o más entradas de la EDT.

1 Gestión del proyecto.

2 Análisis de requisitos.

- 2.1 Análisis de la situación actual de aplicaciones de tutoriales.
- 2.2 Estudio de los servicios básicos a ofrecer.
- 2.3 Estudio de las funciones finales de Tipi-Tapa.
- 2.4 Análisis de las tecnologías de desarrollo.

3 Estudio previo.

- 3.1 Estudio de los diseños modernos de páginas web.
- 3.2 Estudio de aplicaciones similares.
- 3.3 Mejoras propuestas a los sistemas existentes.

4 Adquisición y aprendizaje de las tecnologías a emplear.

- 4.1 Aprendizaje de HTML5.
- 4.2 Aprendizaje de CSS3.
- 4.3 Aprendizaje del Framework elegido.
- 4.4 Instalación y aprendizaje de las herramientas escogidas.

5 Implementación y desarrollo.

- 5.1 Diseño de la interfaz de la aplicación.
- 5.2 Extracción del modelo de dominio.
- 5.3 Diseño e implementación de la base de datos.
- 5.4 Diseño e implementación de la arquitectura del proyecto.
- 5.5 Diseño e implementación de la aplicación.
- 5.6 Implantación en entorno de producción-desarrollo.

6 Control de calidad.

- 6.1 Generación de lista de pruebas unitarias.
- 6.2 Generación y ejecución del plan de pruebas.
- 6.3 Optimización de la aplicación.

7 Documentación.

- 7.1 Memoria del proyecto.
- 7.2 Manual de usuario.

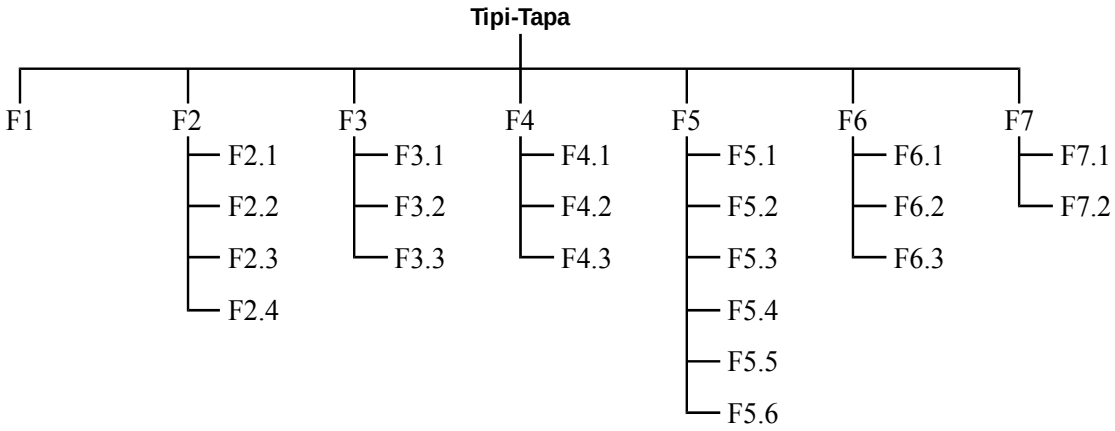


Ilustración 2: Diagrama EDT

1.4.2. Lista de actividades

- **(F1) Gestión del proyecto**

  El desarrollo de software de gran porte requiere una adecuada gestión del proyecto. Hay presupuestos, establecimiento de tiempos de entrega, recursos (lugar de trabajo, insumos, equipamiento) por adquirir. Para su administración se debe tener una clara visión y capacitación en Gestión de Proyectos.

- **(F2) Análisis de requisitos**
  - **(F2.1) Análisis de la situación actual de aplicaciones de tutoriales**

    Investigación de la situación actual de aplicaciones (sean web o no) de tutoriales. La búsqueda de dicha información será a través de Internet, en las páginas web principales de estos programas.

  - **(F2.2) Estudio de los servicios básicos a ofrecer**

    Estudio de las funciones ya implementadas en aplicaciones de tutoriales y examen del denominador común de todas ellas, además de las funcionalidades que todas implementan.

  - **(F2.3) Estudio de las funciones finales de Tipi-Tapa**

    Su desarrollo puede realizarse paralelamente al estudio de los servicios básicos a ofrecer, con lo que se ahorraría tiempo. Trata de la realización de especificaciones necesarias para el diseño de las funciones finales del proyecto en base a la información recopilada de otros programas.

  - **(F2.4) Análisis de las tecnologías de desarrollo**

    Análisis de las tecnologías Web en Java EE más actuales con el fin de encontrar la más idónea para el desarrollo de la aplicación.

- **(F3) Estudio previo**
  - **(F3.1) Estudio de los diseños modernos de páginas web**

    Estudio de algunas de las páginas web modernas más conocidas. No sólo se llevará a cabo un estudio gráfico de dichas pantallas, también se tendrán en cuenta diversos aspectos de calidad del software en páginas Web.
  - **(F3.2) Estudio de aplicaciones similares**

    Está claro que el proyecto no va a cubrir una nueva necesidad de mercado, sólo intentará mejorar lo ya existente o implementar aspectos novedosos que no se dan entre las alternativas actuales. El estudio de aplicaciones como "quehayenlanevera.com" o "hogarutil.com" han sido de gran utilidad a la hora de diseñar Tipi-Tapa.
  - **(F3.3) Mejoras propuestas a los sistemas existentes**

    Divergencias de Tipi-Tapa con programas parecidos ya lanzados al mercado global. No sólo de las nuevas funciones que posea, sino de las mejoras a las funciones ya implementadas por otros programas. Es lo que debe darle competitividad al producto final.
- **(F4) Adquisición y aprendizaje de las tecnologías a emplear**
  - **(F4.1) Aprendizaje de HTML5 y CSS3**

    Búsqueda de documentación y estudio del las últimas versiones del lenguaje HTML y CSS con los que se construirá la base de la interfaz de la aplicación. Será necesario un conocimiento experto de ambas tecnologías.
  - **(F4.2) Aprendizaje del Framework elegido**

    Búsqueda de documentación y estudio de las últimas versiones del Framework seleccionado para el desarrollo de la aplicación. Será necesario un conocimiento bastante amplio de las funcionalidades a utilizar en el desarrollo.
  - **(F4.3) Instalación y aprendizaje de las herramientas escogidas**

    Tanto las herramientas necesarias para albergar la aplicación como las que se necesiten para su desarrollo deberán ser descargadas, instaladas y configuradas correctamente. Previamente se requerirá un proceso de aprendizaje de las mismas.
- **(F5) Implementación y desarrollo**
  - **(F5.1) Diseño de la interfaz de la aplicación**

    Diseño de la parte visual de Tipi-Tapa y sus diferentes secciones. Este proceso se realizará mediante la técnica conocida como "*Wireframing*".

Un wireframe para un sitio web, también conocido como un esquema de página o plano de pantalla, es una guía visual que representa el esqueleto o estructura visual de un sitio web. El wireframe esquematiza el diseño de página u ordenamiento del contenido del sitio web, incluyendo elementos de la interfaz y sistemas de navegación. Usualmente este esquema carece de estilo tipográfico, color o aplicaciones gráficas ya que su principal objetivo reside en la funcionalidad, comportamiento y jerarquía de contenidos. En otras palabras, se enfoca en "qué hace la pantalla, no cómo se ve." Los esquemas pueden ser dibujados con lápiz y papel, como esquemas en una pizarra o pueden ser producidos con medios de diseño de aplicaciones de software libre o comerciales.

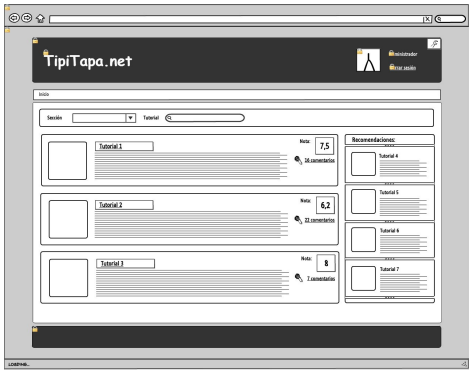


Ilustración 3: Wireframing

Salida:

Documento con el guión de interfaces.

- **(F5.2) Extracción del modelo de dominio**

  El modelo de dominio se crea con el fin de representar el vocabulario y los conceptos clave del dominio del problema. El modelo de dominio también identifica las relaciones entre todas las entidades comprendidas en el ámbito del dominio del problema e identifica sus atributos. Un modelo de dominio que encapsula los métodos dentro de las entidades se asocia más bien con modelos orientados a objetos que son los que se van a emplear en el proyecto.

  Una ventaja importante de un modelo de dominio es que describe y limita el alcance del dominio del problema. El modelo de dominio puede ser usado efectivamente para verificar y validar la comprensión del dominio del proyecto.

  Puede ser tomado como el punto de partida para el diseño del sistema. Cuando se realiza la programación orientada a objetos, el funcionamiento interno del software va a imitar en alguna medida a la realidad, por lo que el mapa de conceptos del modelo de dominio constituye una primera versión del sistema.

  Salida:

  Diagrama en notación UML del modelo de dominio

- **(F5.3) Diseño e implementación de la base de datos**

  Diseño, implementación e implantación de la base de datos que dará soporte

a la aplicación. Para su diseño habrá que fijarse en el modelo de dominio generado en el modelo de dominio y en las tecnologías escogidas y estudiadas.

Salida:

Diagrama de Entidad/Relación de la base de datos y scripts de creación.

- **(F5.4) Diseño e implementación de la arquitectura del proyecto**

  Diseño e implementación de la estructura principal de unidades de desarrollo, paquetes de archivos fuente, ficheros de configuración, etc. sobre la cual se implementará la aplicación.

  Esta arquitectura vendrá definida en su mayor parte por lo aprendido sobre el Framework elegido.

- **(F5.5) Diseño e implementación de la aplicación**

  Diseño y desarrollo de todos los módulos funcionales de los que se componga la aplicación. Esta actividad alberga todo el desarrollo propiamente dicho de Tipi-Tapa.

  Salida:

  Entregable del proyecto que contenga el código fuente.

- **(F5.6) Implantación en entorno de producción-desarrollo**

  Esta actividad se refiere a las acciones a realizar para poder ejecutar la aplicación desarrollada en los entornos de desarrollo y pruebas respectivamente.

- **(F6) Control de calidad**
  - **(F6.1) Generación de lista de pruebas unitarias**

    En programación, una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Luego, con las Pruebas de Integración, se podrá asegurar el correcto funcionamiento del sistema o subsistema en cuestión.

    La idea es escribir casos de prueba para cada función no trivial o método en el módulo, de forma que cada caso sea independiente del resto.

    Salida:

    Listado de pruebas unitarias.

  - **(F6.2) Generación y ejecución del plan de pruebas**

    Listado con el orden de las pruebas de integración a realizar.

    Pruebas integrales o pruebas de integración son aquellas que se realizan en el

ámbito del desarrollo de software una vez que se han aprobado las pruebas unitarias. Únicamente se refieren a la prueba o pruebas de todos los elementos unitarios que componen un proceso, realizada en conjunto, de una sola vez.

Consiste en realizar pruebas para verificar que un gran conjunto de partes de software funcionan juntos.

Las pruebas de integración (algunas veces llamadas integración y testeo I&T) es la fase de la prueba de software en la cual módulos individuales de software son combinados y probados como un grupo. Son las pruebas posteriores a las pruebas unitarias y preceden a las pruebas del sistema.

Salida:

Documento de plan de pruebas

- **(F6.3) Optimización de la aplicación**

  La optimización de software es el proceso de modificación de un software para hacer que algún aspecto del mismo funcione de manera más eficiente y/o utilizar menos recursos (obteniendo un mayor rendimiento). En general, un programa puede ser optimizado para que se ejecute más rápidamente, utilice menos memoria u otros recursos o consuma menos energía.

  Salida:

  Informes de velocidad de páginas de renombre en este ámbito.

- **(F7) Documentación**
  - **(F7.1) Memoria del proyecto**

    Realización de la memoria del proyecto y preparación de la presentación para la defensa del mismo

    Salida:

    Memoria del proyecto y presentación

  - **(F7.2) Manual de usuario**

    Realización del manual de usuario de Tipi-Tapa en libro y en formato pdf.

    Salida:

    Manual de usuario de Tipi-Tapa.

1.5. Planificación del proyecto

En este apartado se llevará a cabo la planificación del proyecto. Se establecerá el orden de las tareas a realizar, el tiempo necesario para efectuarlas, el coste de las mismas, se elaborará un análisis de posibles riesgos a tener en cuenta, un plan de cambios para prever posibles modificaciones y se clarificarán una serie de pautas de calidad y un plan de contingencias frente a posibles adquisiciones.

Una vez realizada esta parte del proyecto se procederá a hacer un seguimiento y control del mismo.

1.5.1. Plan temporal

1.5.1.1. Planificación temporal inicial

A continuación, la siguiente tabla muestra la lista de tareas que se realizarán durante el desarrollo del proyecto junto con su fecha de comienzo y fin y el trabajo que se necesitará para realizarlas según lo planificado.

No obstante, la planificación temporal puede sufrir variaciones dando lugar a una posterior redistribución de tareas para ampliar los plazos de finalización del proyecto.

A continuación se muestran un resumen con la planificación de las tareas del EDT:

Nombre	Inicio	Fin	Duración
(F1) Gestión del proyecto	25. sep. 2013	1. oct. 2013	20h
(F2.1) Análisis de la situación actual de aplicaciones de tutoriales	25. sep. 2013	25. sep. 2013	4h
(F2.2) Estudio de los servicios básicos a ofrecer	26. sep. 2013	26. sep. 2013	3h
(F2.3) Estudio de las funciones finales de Tipi-Tapa	26. sep. 2013	26. sep. 2013	4h
(F2.4) Análisis de las tecnologías de desarrollo	27. sep. 2013	28. sep. 2013	8h
(F3.1) Estudio de los diseños modernos de páginas web	28. sep. 2013	28. sep. 2013	4h
(F3.2) Estudio de aplicaciones similares	28. sep. 2013	29. sep. 2013	4h
(F3.3) Mejoras propuestas a los sistemas existentes	28. sep. 2013	29. sep. 2013	4h
(F4.1) Aprendizaje de HTML5 y CSS3	29. sep. 2013	4. oct. 2013	28h
(F4.2) Aprendizaje del Framework elegido	4. oct. 2013	11. oct. 2013	40h
(F4.3) Instalación y aprendizaje de las herramientas escogidas	29. sep. 2013	4. oct. 2013	28h
(F5.1) Diseño de la interfaz de la aplicación	11. oct. 2013	17. oct. 2013	36h
(F5.2) Extracción del modelo de dominio	18. oct. 2013	19. oct. 2013	12h
(F5.3) Diseño e implementación de la base de datos	19. oct. 2013	23. oct. 2013	24h
(F5.4) Diseño e implementación de la arquitectura del proyecto	19. oct. 2013	5. nov. 2013	100h
(F5.5) Diseño e implementación de la aplicación	6. nov. 2013	22. may. 2014	1000h
(F5.6) Implantación en entorno de producción-desarrollo	23. may. 2014	26. may. 2014	8h
(F6.1) Generación de lista de pruebas unitarias	5. nov. 2013	5. nov. 2013	4h
(F6.2) Generación y ejecución del plan de pruebas	26. may. 2014	3. jun. 2014	48h
(F6.3) Optimización de la aplicación	26. may. 2014	30. may. 2014	16h
(F7.1) Memoria del proyecto	3. jun. 2014	22. jun. 2014	120h
(F7.2) Manual de usuario	22. jun. 2014	25. jun. 2014	20h

Tabla 1: Resumen del plan

1.5.1.2. Diagrama de Gant

Este diagrama nos muestra el tiempo de dedicación previsto para las tareas del proyecto en relación al tiempo total empleado. A pesar de que en principio no indica las dependencias existentes entre actividades, la posición de cada tarea hace posible la identificación de dichas relaciones.

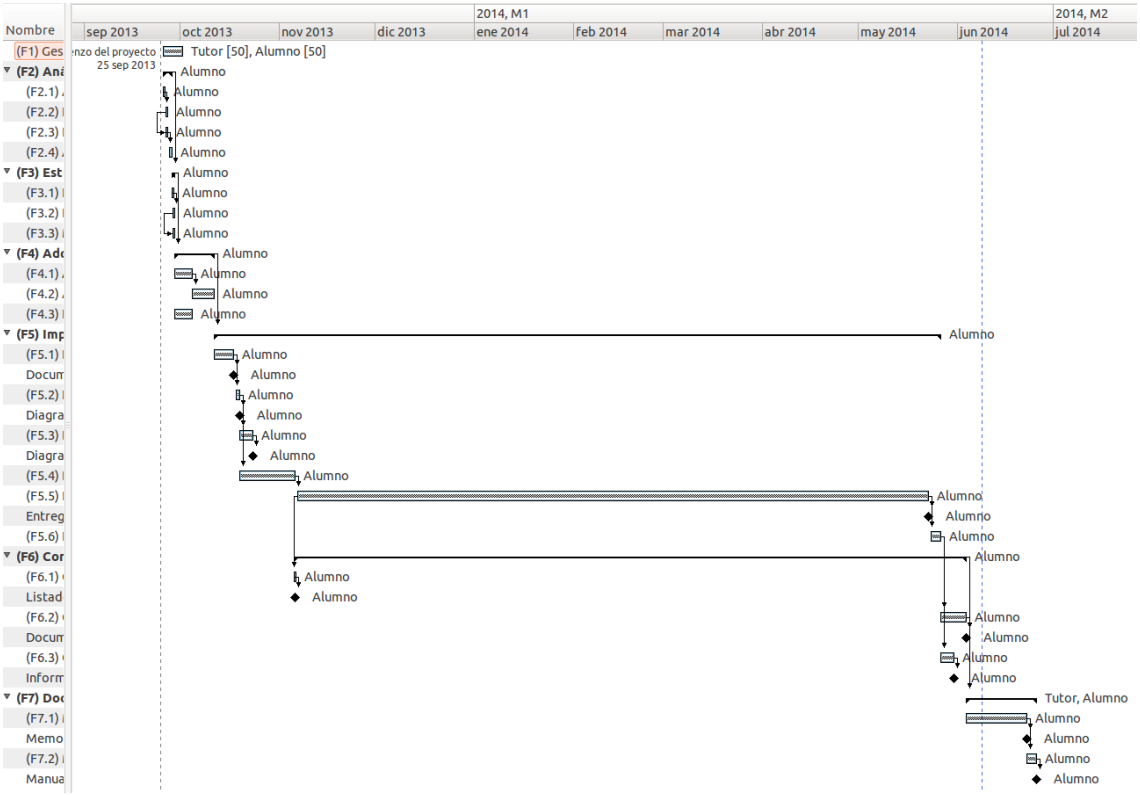


Ilustración 4: Diagrama de Gantt (Resumen)

1.5.1.3. Hitos y entregables

- Documento con el guión de interfaces.
- Diagrama en notación UML del modelo de dominio.
- Diagrama de Entidad/Relación de la base de datos y scripts de creación.
- Entregable del proyecto que contenga el código fuente.
- Listado de pruebas unitarias.
- Documento de plan de pruebas.
- Informes de velocidad de páginas de renombre en este ámbito.

- Memoria del proyecto y presentación.
- Manual de usuario de Tipi-Tapa.

1.5.1.4. Fechas de inicio/fin de proyecto

Se estima que la realización del proyecto durará del día **25 de Septiembre de 2013** al **25 de Junio de 2014** con un total de **1.535 horas** dedicadas. Esta planificación está adaptada con la finalidad de compatibilizar el desarrollo del proyecto con la vida laboral diaria.

1.5.1.5. Bases de estimación

A continuación se detallan aquellos elementos que se identifican como variables a tener cuenta en el cálculo de la estimación de tiempos de las tareas.

- Personas físicas involucradas en el proyecto:

  Serán **dos personas**. El alumno autor del proyecto y el tutor, encargado de guiar al alumno en la gestión del mismo y de revisar para obtener un correcto resultado final de la aplicación y de la documentación generada .

- Calendario:

  La formación de un calendario de horas disponibles al desarrollo del proyecto se hace harto difícil dados los posibles imprevistos originados en el ámbito laboral y personal. Dado que éste tipo de variaciones ya han sido consideradas debidamente en el análisis de riesgos y se tratan de variables impredecibles no se tendrán en cuenta.

  - Del 25 de Octubre 2013 al 31 de Mayo 2014
    - Horas dedicadas semanales: 40h
    - Vacaciones:
      - Del 16 de Diciembre 2013 al 3 de Enero 2014
      - Del 17 de Abril 2014 al 21 de Abril 2014
  - Del 1 de Junio 2014 al 31 de Junio 2014
    - Horas dedicadas semanales: 48h
    - Vacaciones:
      - 14 de Junio 2014

1.5.2. Plan de costes (Análisis de viabilidad económica)

El análisis económico del proyecto consistirá en señalar los costes de desarrollo del mismo y compararlos con los beneficios que producirá una vez desarrollado.

El plan se ha desglosado en los siguientes apartados:

1. Inmovilizado inmaterial
2. Inmovilizado material
3. Mano de obra
4. Gastos imputables al proyecto

1.5.2.1. Inmovilizado inmaterial

El proyecto en su totalidad va a ser desarrollado mediante el uso exclusivo de herramientas libres. Esto implica que en ningún momento habrá que realizar pago alguno de licencias para el uso de las mismas.

En la siguiente tabla se muestran las licencias y costes de las herramientas utilizadas:

Inmovilizado inmaterial	**Licencia**	**Web de descarga**	**Coste**
LibreOffice	LGPLv3	http://es.libreoffice.org/	0,00 €
GIMP	GNU GPL	http://www.gimp.org.es/	0,00 €
StarUML	GNU GPL	http://staruml.sourceforge.net/	0,00 €
Planner	GPLv2+	https://wiki.gnome.org/Apps/Planner	0,00 €
Apache	Apache v2	http://httpd.apache.org/	0,00 €
Jboss	LGPLv2.1	http://www.jboss.org/products/webserver/	0,00 €
sshd	GNU GPL	Repositiorios ubuntu	0,00 €
subversion	GNU GPL	Repositiorios ubuntu	0,00 €
Eclipse	EPL	http://www.eclipse.org/	0,00 €
Firefox	MPL	http://www.mozilla.org/	0,00 €
Chrome	Freeware	http://www.google.es/intl/es/chrome/	0,00 €
Opera	Freeware	http://www.opera.com/	0,00 €
Safari	Freeware	https://www.apple.com	0,00 €
Internet Explorer	Freeware	http://windows.microsoft.com/es-es/internet-explorer/download-ie	0,00 €
		TOTAL	**0,00 €**

Tabla 2: Inmovilizado inmaterial

1.5.2.1. Inmovilizado material

Los únicos objetos cuyas amortizaciones son impugnables al proyecto son ordenadores empleados en la realización del mismo y en los dispositivos móviles usados en las pruebas.

El coste de horario de cada una de estas propiedades se ha desglosado en la siguiente tabla para poder imputarlo correctamente al proyecto. La estimación horaria se ha basado en el tiempo que se tardará en realizar las tareas que implican el uso de dichos elementos. El cálculo del coste es paralelo al realizado en el caso del inmovilizado inmaterial y se adjunta en la siguiente tabla:

Inmovilizado material	Coste	Amortización	Coste anual	Vida útil anual	Coste/Ud. de vida	Uds. asignables	Coste en proyecto
PC Sobremesa	500,00 €	7 años	71,43 €	2.080 h	0,03 €/h	193 h	5,79 €
Portátil	2.332,45 €	2 años	1.166,23 €	2.080 h	0,56 €/h	1.152 h	645,12 €
Smartphone	439,00 €	1 año	439,00 €	2.080 h	0,21 €/h	95 h	19,95 €
Tablet	460,68 €	1 año	460,68 €	2.080 h	0,22 €/h	95 h	20,90 €
						TOTAL	**691,76 €**

Tabla 3: Inmovilizado material

1.5.2.1. Mano de obra

Como el tiempo empleado en el desarrollo del proyecto se realiza fuera del horario laboral no se genera coste en mano de obra alguno. Se calculará una estimación de coste de mano de obra puesto que serán necesarias para un posterior estudio de la rentabilidad del proyecto o en la estimación del valor del mismo.

La estimación del coste de la mano de obra se ha realizado en función del convenio colectivo de oficinas y despachos de Bizkaia que no se encuentra en vigencia en la actualidad.

Dicho convenio podemos encontrarlo en :

http://www.comfia.net/archivos/OyDBizkaia0912_Castellano.pdf

Como se puede leer dicta que el salario anual base para un analista programador es de 24.482,94 € para una jornada laboral de 1.742 horas como máximo.

Mano de obra	Coste anual	Coste por hora	Horas de proyecto	Coste en proyecto
Lander Laparra	24.482,94 €	14,05 €	1.535 h	21.573,66 €
			TOTAL	**21.573,66 €**

Tabla 4: Mano de obra

1.5.2.2. Otros gastos

En este apartado se recogen los gastos directamente imputables al proyecto como la impresión de la memoria.

Gastos del proyecto	Coste/Ud. De vida	Uds. asignables	Coste en proyecto
Impresión memoria	0,10 €/pág.	200 págs.	20,00 €
		TOTAL	**20,00 €**

Tabla 5: Otros gastos

1.5.2.1. Presupuesto del proyecto

Finalmente, se recogen todos los gastos analizados anteriormente en una misma tabla. El total que figura en dicha tabla, recoge la estimación del plan de costes del proyecto y no considera el beneficio que pueda obtenerse de la venta del mismo.

Gastos del proyecto	Coste
Amortizaciones inmateriales	0,00 €
Amortizaciones materiales	691,76 €
Mano de obra imputable	21.573,66 €
Otros gastos	20,00 €
TOTAL	**22.285,42 €**

Tabla 6: Presupuesto del proyecto (I)

Presupuesto	Coste
Costo del proyecto	22.285,42 €
Costos generales (10%)	2.228,54 €
Beneficio (15%)	3.342,81 €
Subtotal	**27.856,78 €**
IVA (18%)	5.014,22 €
TOTAL	**32.870,99 €**

Tabla 7: Presupuesto del proyecto (II)

Por tanto, el presupuesto asciende a treinta y dos mil ochocientos setenta euros con noventa y nueve céntimos.

1.5.3. Calidad

1.5.3.1. Introducción

Para realizar una correcta planificación de calidad es necesario identificar las pautas relevantes para el proyecto, así como buscar la forma de satisfacerlas evitando posibles consecuencias. Una sobre-asignación de trabajo o una ejecución apresurada de las actividades relacionadas con la calidad puede hacer que se pasen por alto fallos o errores.

La calidad mínima en este proyecto serán los requisitos definidos en la planificación y definición del alcance, tras la realización una correcta comprensión, evaluación y definición de las expectativas del mismo, cumpliendo con los plazos y costes acordados.

Como se ha definido con anterioridad, será necesaria una prevención frente a riesgos no identificados de un 14% del cómputo global de horas.

1.5.3.2. Planificación de calidad del proyecto

La calidad del proyecto vendrá marcada por la planificación y gestión que se haga del mismo. Se seguirá una gestión de calidad centralizada, haciendo referencia a las cuestiones que afectan a dicha calidad únicamente en este apartado de la planificación, que abarcará además todo lo necesario para la consecución de un proyecto de garantías.

1.5.3.3. Planificación de calidad del producto

Según la norma ISO 9000:2000, la planificación de la calidad es la parte de la gestión de la calidad enfocada al establecimiento de los objetivos de la calidad y a la especificación de los procesos operativos necesarios y de los recursos relacionados para cumplir los objetivos de calidad.

1.5.3.3.1. Rol de la planificación

El rol de esta planificación será establecer una forma de trabajo para asegurar la calidad del software desarrollado durante el desarrollo y al final del mismo. Se establecerán unos requerimientos de obligado cumplimiento y unas pautas a seguir para su comprobación de manera que el índice de calidad del producto final sea lo más elevado posible.

1.5.3.3.2. Requerimientos de la Calidad de Software

- Requerimientos de interfaz:
  - La interfaz de la aplicación debe visualizarse correctamente en dispositivos de distinta resolución de pantalla. Se entiende por una visualización correcta que todas las opciones visibles en la pantalla permanezcan así en cualquier resolución (no habrá pérdidas de funcionalidad en resoluciones pequeñas).
  - Los elementos de la interfaz sobre los que se puedan realizar eventos de clic con el ratón deberán mostrarse lo suficientemente grandes en dispositivos táctiles. De esta forma dichas acciones podrán llevarse a cabo fácilmente con los dedos.
  - Validaciones:
    - Los campos obligatorios deberán mostrar una validación cuando un usuario envíe un formulario con ellos en blanco.
    - Las fechas deberán mostrar una validación cuando un usuario envíe un formulario con un valor que no sea una fecha en ellos.
    - Los campos numéricos deberán mostrar una validación cuando un usuario envíe un formulario con un valor no numérico en ellos.
  - Feedback de la aplicación:
    - Toda acción que el usuario realice con la aplicación deberá tener un feedback inmediato de la misma para que el usuario entienda que ha realizado la acción correctamente.
    - En caso de haber tiempos de espera se mostrará un spinner indicando que la acción ejecutada está en proceso.
    - Los mensajes de feedback tendrán diferentes niveles: verde cuando todo ha ido correctamente, azul cuando se trate de información relevante para el usuario y rojo cuando haya habido algún error o salten las validaciones.
    - Los errores que se muestren al usuario no incluirán información incomprensible para el mismo. Se mostrará un mensaje predefinido.
    - Todos los campos de texto deberán informar del número de caracteres que pueden recibir como máximo, no pudiendo introducirse más de los especificados.
- Requerimientos de la seguridad:
  - En ningún caso se permitirá que un usuario pueda realizar acciones de un rol

que no sea el suyo propio.

- Aparte de las validaciones realizadas en la parte de presentación se deberán realizar las propias en el servidor con el fin de evitar ataques que causen un funcionamiento incorrecto de la aplicación.
- No debe permitirse la inyección de código SQL. Todos los parámetros se introducirán como tales en las operaciones de base de datos y no a la hora de formar las querys.

- Requerimientos de rendimiento:
  - Todas las imágenes de los contenidos irán en peticiones separadas siendo así identificables por los navegadores y pudiendo ser cacheadas.
  - Las pantallas mostradas deben ser lo suficientemente ligeras para no tener tiempos de carga pesados.
- Requerimientos generales:
  - No deben darse errores funcionales.
  - No deben darse errores en operaciones contra la base de datos originados por el contenido de los datos manejados.
  - La gestión de errores imprevisibles como son los errores de conexión deben estar gestionados correctamente.

1.5.3.3.3. Preparación de un plan de Calidad de Software

El desarrollo de la aplicación se dividirá en diferentes módulos. Estos módulos pueden estar compuestos por una o varias pantallas de interfaz y se agruparán normalmente por caso de uso.

Para cada uno de estos módulos se genera un análisis, planificación y diseño específicos de la misma forma que se generará un plan de pruebas propio.

Este plan de pruebas se validará una vez terminado el desarrollo de dicho módulo justo después de haber realizado con un resultado satisfactorio todas y cada una de las pruebas unitarias.

Cada vez que una prueba del plan tenga un resultado no satisfactorio se realizarán las correcciones necesarias y se empezará por pasar de nuevo las pruebas unitarias y el plan de pruebas del módulo. Estas operaciones se realizaran tantas veces como resulten necesarias hasta que el plan de pruebas se complete satisfactoriamente.

1.5.3.3.4. Implementación del plan de Calidad de Software

Como se ha comentado en el apartado anterior cada módulo de aplicación tendrá su propio plan de pruebas específico que puede consultarse en los puntos de este

documento dedicados a cada módulo de la aplicación.

Las pruebas unitarias serán las siguientes:

ID	PRUEBA A REALIZAR
1	Prohibidos los SELECT *
2	Añadir registro a la BBDD (se debe comprobar la inserción correcta en la BBDD).
3	Búsqueda y Consulta (se debe comprobar el resultado de la búsqueda y la visualización correcta de la misma y del detalle de los registro en modo consulta).
4	Navegación por los botones de la pantalla (Si tienen que cargar otras páginas, inicio, etc…).
5	Eliminar registro, comprobar que la baja es de carácter lógico y se realiza correctamente.
6	Comprobar que se muestran los mensajes correspondientes a la hora de la realización de las operaciones: - Grabaciones correctas. - Confirmaciones previas a los borrados. - Mensajes de confirmación cuando haya cambios de páginas sin guardado de los cambios. …
7	Comprobar que todos los módulos se visulicen y funcionen correctamente en Internet Explorer, Chrome y Firefox
8	Comprobar que en los mantenimientos con la opción de añadir, se muestra el mensaje informativo y la pantalla muestra el detalle del registro introducido en modo consulta.
9	Comprobar que todos los campos tienen que estar relacionados con su etiqueta a través de la propiedad for. <label for="dni">DNI</label> <input id="dni" value=""…>
10	
11	Comprobar que no se puede salir de ninguna página si hay cambios sin guardar, la aplicación debe preguntar.
12	Comprobar que la pantalla se redimensiona bien cuando cambia la resolución.
13	Se utilizan las validaciones para los campos de los formularios definidas.
14	Comprobar que se controlan todos los campos obligatorios, que están identificados y se generan mensajes de aviso si no están completados.
15	Los errores producidos sobre los formularios se indican en los campos implicados, como ayuda al mensaje de error.
16	Comprobar que se ha Utilizado la tercera persona a la hora de escribir los textos de los mensajes.
17	Comprobar la alineación de los resultados en las Tablas de Resultados (que muestran lo obtenido en las búsquedas). Los textos a la izquierda y los datos numéricos a la derecha.
18	Comprobar que los rastros de migas están correctos en cada página.
19	Comprobar que no se pueden realizar cambios mientras se están realizando acciones bloqueantes.
20	Comprobar las ordenaciones de las columnas de las tablas resultantes de las búsquedas.

Tabla 8: Pruebas unitarias

1.5.4. Planificación de la gestión de riesgos

El análisis de riesgos es el proceso que permite la identificación de las amenazas que pueden afectar al desarrollo del proyecto y determinar el impacto o grado de perjuicio que pueden ocasionar. Implica analizar las amenazas y vulnerabilidades que pueden darse y se valoran en términos de probabilidad de ocurrencia y gravedad de impacto sobre el proyecto.

1.5.4.1. Metodología

A continuación se establece una escala (de menor a mayor gravedad) para valorar el impacto que pueden causar los riesgos que se describen a continuación, en caso de darse:

- **Muy grave:** De consecuencias fatales para el desarrollo del proyecto. Requeriría replantearse incluso si merece la pena continuar con el mismo.
- **Grave:** Problemas que podrían imposibilitar terminar el proyecto en el plazo previsto. Requeriría realizar una profunda replanificación ampliando el número de horas a dedicar o el trabajo a realizar.
- **Perjudicial:** Pequeños retrasos relativamente sencillos de solventar, aunque pueden resultar peligrosos si no se identifican y subsanan a tiempo.
- **Leve:** Incidencia de poca importancia que en ningún caso compromete el desarrollo normal del proyecto.

Será importante también clasificar los riesgos según las siguientes características:

- **Controlado / No controlado**
- **Previsible / No previsible**
- **De origen interno / externo**
- **Impacto**
- **Plan de contingencia**

1.5.4.2. Identificación y análisis cualitativo de riesgos

Riesgo: **Retardo en el aprendizaje de nuevas tecnologías**			
Probabilidad: 30%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Puede influir en la consecución del proyecto			
Plan de contingencias:	Para el desarrollo del proyecto es necesario tener un conocimiento pleno de las tecnologías empleadas tanto para el soporte como para la implementación del mismo para que futuras tareas sean realizadas de manera correcta. Así se evitaría una desviación aún más importante.		

Tabla 9: Análisis de riesgos (I)

Riesgo: **Pérdida de conexión con el servidor o caída del mismo**			
Probabilidad: 40%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Imposibilidad de conexión con la aplicación de forma remota			
Plan de contingencias:	Se contará con un equipo portátil con todos los servicios necesarios instalados y configurados de tal forma que la aplicación sea funcional en el mismo entorno de desarrollo en caso de ser necesario.		

Tabla 10: Análisis de riesgos (II)

Riesgo: **Elección equivocada de las herramientas de trabajo**			
Probabilidad: 10%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Es posible que con alguna de las tecnologías elegidas no sea posible cumplir los objetivos planteados o dar a la aplicación la funcionalidad deseada.			
Plan de contingencias:	Realizar un estudio profundo sobre las diferentes tecnologías y tener planteadas tecnologías alternativas para comenzar a utilizarlas lo antes posible.		

Tabla 11: Análisis de riesgos (III)

Riesgo: **Estancamiento en la codificación**			
Probabilidad: 50%	Controlado: SI	Previsible: NO	Origen: Interno
Impacto: Podría llegar a darse algún problema de difícil solución que paralizaría momentáneamente el avance del proyecto.			
Plan de contingencias: Se ha dejado de margen el 14% de las horas planificadas para posibles situaciones como ésta.			

Tabla 12: Análisis de riesgos (IV)

Riesgo: **Baja del desarrollador de la aplicación**			
Probabilidad: 8%	Controlado: NO	Previsible: NO	Origen: Interno
Impacto: Podría suceder que por enfermedad o accidente, el desarrollador del proyecto quede incapacitado durante un período de tiempo provocando la paralización del proyecto.			
Plan de contingencias: No existe.			

Tabla 13: Análisis de riesgos (V)

Riesgo: **Análisis erróneo o planificación muy optimista**			
Probabilidad: 60%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Una mala planificación podría acarrear retrasos en el desarrollo del proyecto.			
Plan de contingencias: Modificar la planificación alargando la duración del proyecto o disminuyendo el alcance del mismo.			

Tabla 14: Análisis de riesgos (VI)

Riesgo: **Pérdida de los datos del proyecto por fallo del hardware o software maligno**			
Probabilidad: 10%	Controlado: NO	Previsible: NO	Origen: Interno
Impacto: Debido a un virus o a un fallo en los componentes del ordenador en el que se desarrolla el proyecto podría ocasionar una pérdida total o parcial de los datos del mismo.			
Plan de contingencias: Tener copias de seguridad del proyecto en distintos dispositivos e instalar un antivirus y cortafuegos en el sistema.			

Tabla 15: Análisis de riesgos (VII)

1.5.4.3. Seguimiento y control de riesgos

El seguimiento y control de riesgos será semanal y su principal objetivo será encontrar los nuevos riesgos que han surgido (o se han eliminado) debido a los cambios de escenario dentro del proyecto.

Durante el seguimiento y control de riesgos se comprobará que no están apareciendo y que los que puedan prevenirse están siendo tratados debidamente y se han tomado las medidas correspondientes.

1.5.5. Adquisiciones

Para poder realizar una buena gestión de adquisiciones es necesario identificar qué necesidades del proyecto se pueden alcanzar mejor a través de la adquisición de productos y servicios.

La decisión de adquisición se debe tomar siguiendo una estrategia de selección de fuentes que se centre en investigar y analizar los pros y los contras de la externalización del servicio y la revisión de las funciones propias para conseguir el máximo beneficio.

Las adquisiciones pueden ser librerías y componentes creados por otros desarrolladores con licencias abiertas como pueden ser MIT, Apache u otras similares.

En este caso las pautas a seguir se basarán en la inclusión de las licencias de uso donde sea pertinente. Y una mención a los creadores (sea requerida por la licencia o no) en un lugar de la aplicación destinada a tal efecto.

2. ANÁLISIS DE REQUISITOS

El propósito de este proceso es conseguir la especificación detallada del sistema de información, a través de un catálogo de requisitos y una serie de modelos que cubran las necesidades de información de los usuarios para los que se desarrollará el sistema de información y que serán la entrada para el proceso de Diseño del Sistema de Información.

Se recogen de forma detallada los requisitos funcionales que el sistema de información debe cubrir, catalogándolos. Eso permite hacer la traza a lo largo de los procesos de desarrollo. Además se identifican los requisitos no funcionales del sistema, es decir, las facilidades que ha de proporcionar el sistema; las restricciones a las que estará sometido en cuanto a rendimiento, frecuencia de tratamiento, seguridad, etc.

En esta fase también se realiza un análisis preliminar sobre las pantallas que dispondrá la aplicación; puede consultarse en el apartado de anexos del documento.

2.1. Casos de uso

Los casos de uso evitan la jerga técnica, prefiriendo la lengua del usuario final o del experto del campo del saber al que se va a aplicar.

Cada caso de uso se centra en describir cómo alcanzar una única meta o tarea de negocio. Desde una perspectiva tradicional de la ingeniería de software, un caso de uso describe una característica del sistema.

Los casos de uso pretenden ser herramientas simples para describir el comportamiento del software o de los sistemas. Un caso de uso contiene una descripción textual de todas las maneras que los actores previstos podrían trabajar con el software o el sistema. Los casos de uso no describen ninguna funcionalidad interna (oculta al exterior) del sistema, ni explican cómo se implementará. Simplemente muestran los pasos que el actor sigue para realizar una operación.

De esta forma este documento pretende esquematizar la toma de requisitos separando una a una sus diferentes actividades.

2.1.1. Diagrama general

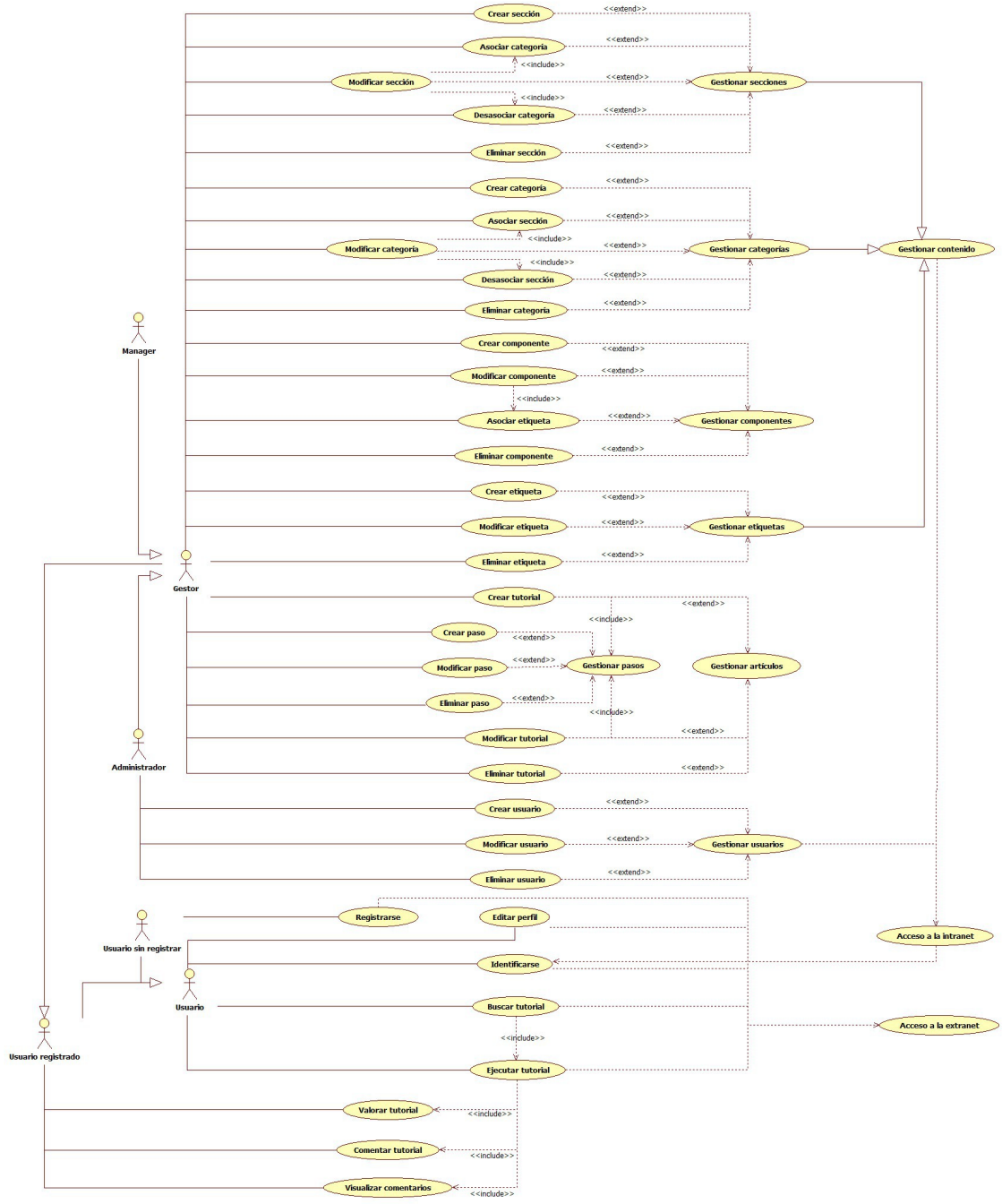


Ilustración 5: Casos de uso - Diagrama general

2.1.2. Diagrama de gestión de usuarios

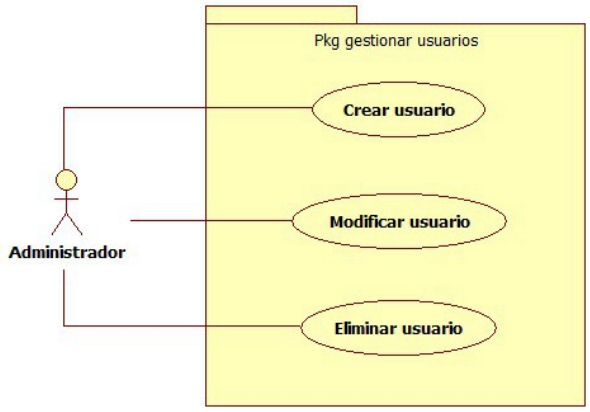


Ilustración 6: Casos de uso - Diagrama de gestión de usuarios

Flujos de eventos

Crear usuario

Flujo de eventos principal:

El caso de uso empieza cuando el administrador pulsa el botón de crear nuevo usuario. Acto seguido deberá rellenar un formulario que contiene los datos del usuario a crear, tales como el nombre de usuario, la contraseña, rol en el sistema, nombre y apellidos, foto, etc.

Una vez el administrador haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo, y en caso contrario se enviará la información al servidor para que registre el nuevo usuario en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador ya había modificado el formulario. En caso afirmativo se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos. Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla. Si se han realizado cambios y no se decide continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de usuario.

Flujo de eventos excepcional:

El administrador puede cancelar un alta de usuario pulsando el botón cancelar, terminando de esta forma el caso de uso.

Modificar usuario

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de un usuario se entra al detalle del mismo con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos del usuario que se puedan modificar, tales como el nombre de usuario, la contraseña, rol en el sistema, nombre y apellidos, foto, etc.

Una vez el administrador haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que se han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. Si son correctos se enviará la información al servidor para que se registren los cambios del usuario editado en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador ya había modificado el formulario. En caso afirmativo se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle de usuario.

Flujo de eventos excepcional:

El administrador puede cancelar las modificaciones de un usuario pulsando el botón cancelar, terminando de esta forma el caso de uso.

Eliminar usuario

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de usuarios se selecciona alguno o algunos de ellos y se pulsa el botón de eliminar. En este momento se borran del sistema tanto los usuarios seleccionados como sus valoraciones de tutoriales.

2.1.3. Diagrama de gestión de secciones

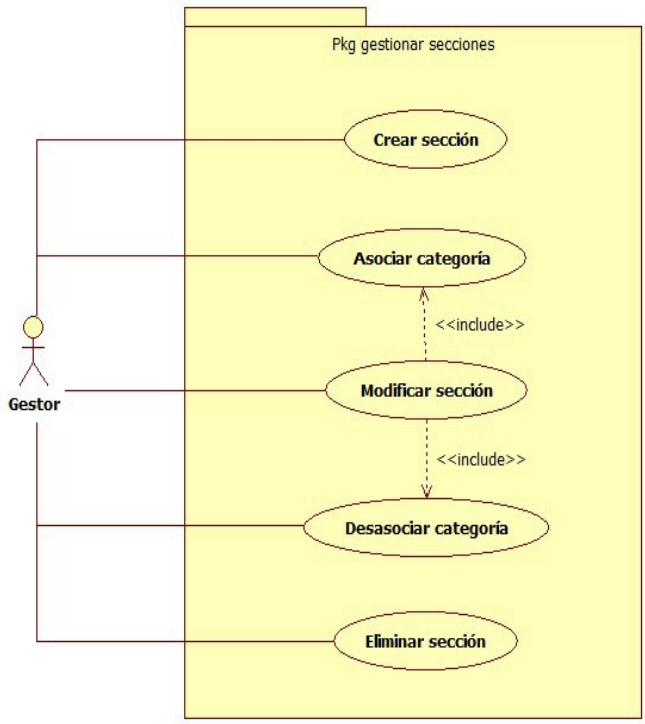


Ilustración 7: Casos de uso - Diagrama de gestión de secciones

Flujos de eventos

Crear sección

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nueva sección. Acto seguido deberá rellenar un formulario que contiene los datos de la sección a crear tales como el nombre de la sección, la descripción, la foto, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que registre la nueva sección en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los

datos introducidos.

Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla. Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de sección.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de sección pulsando el botón cancelar, terminando de esta forma el caso de uso.

Modificar sección

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de una sección se entra al detalle de la misma con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos de la sección que se puedan modificar, tales como el nombre de la sección, la descripción, la foto, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que registren los cambios de la sección editada en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla. Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle de sección.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de una sección pulsando el botón cancelar, terminando de esta forma el caso de uso.

El administrador/manager puede navegar a la pantalla de búsqueda de categorías en modo selección al pulsar el botón de “asociar categoría”. Si se han realizado cambios

antes de navegar a esta pantalla se obligará al usuario mediante un diálogo a decidir entre guardar los cambios o descartarlos. Desde la pantalla de categorías tendremos un botón para asociar las categorías a la sección. Nos retornará a la pantalla de secciones con la información de categorías asociadas a la sección actualizada en la base de datos.

Asociar categoría

Flujo de eventos principal:

El administrador/manager tendrá la opción (desde el caso de uso de modificar una sección) de asociar categorías. Al pulsar un botón a tal efecto se visualizará un listado con las categorías disponibles donde podrá seleccionar y deseleccionar aquellas que desee.

Desasociar categoría

Flujo de eventos principal:

El administrador/manager tendrá la opción (desde el caso de uso de modificar una sección) de desasociar categorías. Las categorías asociadas a la sección se visualizarán en un combo. Al pulsar el botón de desasociar categoría, la categoría seleccionada en el combo se desasociará de la sección que se esté modificando.

Eliminar sección

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de secciones se selecciona alguna o varias de ellas y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación se cerrará el diálogo sin llegar a eliminar nada. En caso contrario al pulsar el botón de continuar se borran del sistema las secciones seleccionadas.

2.1.4. Diagrama de gestión de categorías

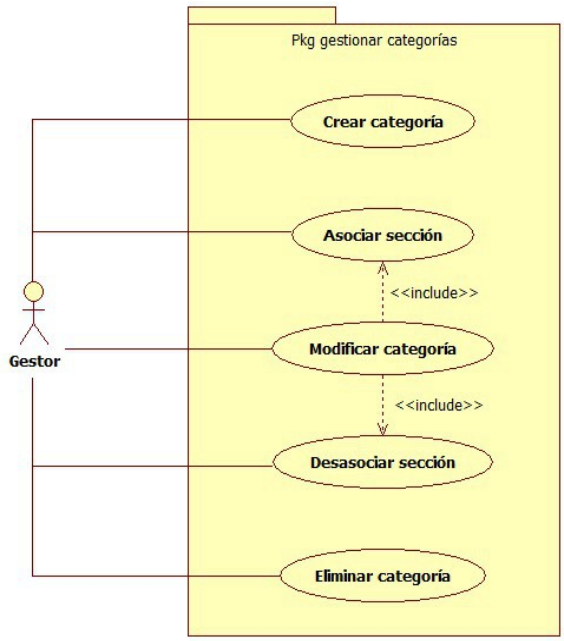


Ilustración 8: Casos de uso - Diagrama de gestión de categorías

Flujos de eventos

Crear categoría

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nueva categoría. Acto seguido deberá rellenar un formulario que contiene los datos de la categoría a crear, tales como el nombre de la categoría, la descripción, la foto, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que registre la nueva categoría en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En caso afirmativo se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla.

Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de categoría.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de categoría pulsando el botón cancelar, terminando de esta forma el caso de uso.

Modificar categoría

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de una categoría se entra al detalle de la misma con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos de la categoría que se puedan modificar, tales como el nombre de la sección, la descripción, la foto, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registren los cambios de la categoría editada en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aun así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla. Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle de categoría.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de una categoría pulsando el botón cancelar, terminando de esta forma el caso de uso.

Asociar sección

Flujo de eventos principal:

El administrador/manager tendrá la opción (desde el caso de uso de modificar una categoría) de asociar secciones. Al pulsar un botón a tal efecto se visualizará un listado con las secciones disponibles donde podrá seleccionar y deseleccionar aquellas que desee.

Desasociar sección

Flujo de eventos principal:

El administrador/manager tendrá la opción (desde el caso de uso de modificar una categoría) de desasociar secciones. Las secciones asociadas a la categoría se visualizarán en un combo. Al pulsar el botón de desasociar sección, la sección seleccionada en el combo se desasociará de la categoría que se esté modificando.

Eliminar categoría

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de categorías se selecciona alguna o varias de ellas y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación se cerrará el diálogo sin llegar a eliminar nada. En caso contrario al pulsar el botón de continuar se borran del sistema las categorías seleccionadas.

2.1.5. Diagrama de gestión de componentes

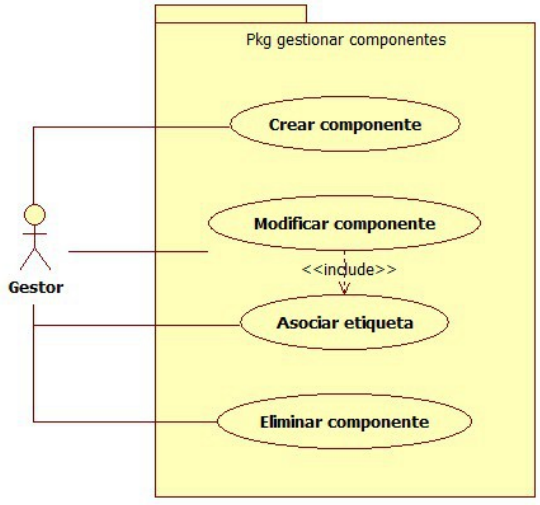


Ilustración 9: Casos de uso - Diagrama de gestión de componentes

Flujos de eventos

Crear componente

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nuevo componente. Acto seguido deberá rellenar un formulario que contiene los datos del componente a crear, tales como el nombre del componente, la descripción, la foto, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que registre el nuevo componente en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar, se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla. Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de componente.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de componente pulsando el botón cancelar, terminando de esta forma el caso de uso.

Modificar componente

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de un componente se entra al detalle del mismo con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos del componente que se puedan modificar, tales como el nombre del componente, la descripción, la foto, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se

mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registren los cambios del componente editado en la base de datos acabando aquí el caso de uso.

- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla. Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle del componente.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de un componente pulsando el botón cancelar, terminando de esta forma el caso de uso.

El administrador/manager puede navegar a la pantalla de búsqueda de categorías en modo selección al pulsar el botón de "asociar categoría". Si se han realizado cambios antes de navegar a esta pantalla se obligará al usuario mediante un diálogo a decidir entre guardar los cambios o descartarlos. Desde la pantalla de categorías tendremos un botón para asociar las categorías al componente. Nos retornará a la pantalla de componentes con la información actualizada en la base de datos.

El administrador/manager puede navegar a la pantalla de búsqueda de etiquetas en modo selección al pulsar el botón de "asociar etiqueta". Si se han realizado cambios antes de navegar a esta pantalla se obligará al usuario mediante un diálogo a decidir entre guardar los cambios o descartarlos. Desde la pantalla de etiquetas tendremos un botón para asociar las etiquetas al componente. Nos retornará a la pantalla de componentes con la información actualizada en la base de datos. Una vez asociadas las etiquetas podremos cambiar el valor asociado al componente desde la pantalla de detalle.

Asociar etiqueta

Flujo de eventos principal:

El caso de uso empieza cuando el usuario pulsa el botón de modificación de etiquetas. En este momento se abre una lista editable con las etiquetas a asociar al componente seleccionado. Aquellas etiquetas sin valor no estarán asociadas y en caso contrario sí lo estarán.

Llegados al punto en el que el usuario haya rellenado las etiquetas que desea asociar tendrá dos opciones: asociar y cancelar. Si selecciona cancelar saldrá de la ventana sin realizar ninguna acción, mientras que pulsando asociar se guardaran los cambios en

base de datos.

Eliminar componente

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de componentes se selecciona alguno o varios de ellos y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación se cerrará el diálogo sin llegar a eliminar nada. En caso contrario al pulsar el botón de continuar se borrarán del sistema los componentes seleccionados.

2.1.6. Diagrama de gestión de etiquetas

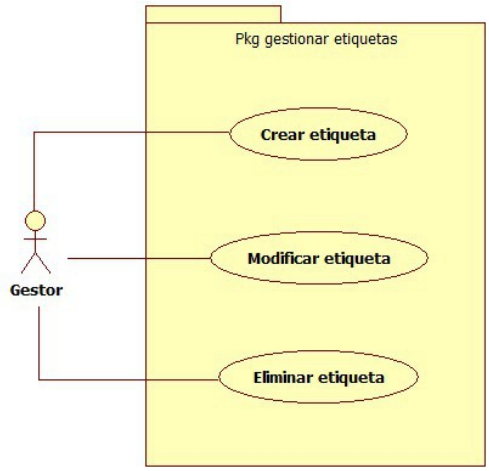


Ilustración 10: Casos de uso - Diagrama de gestión de etiquetas

Flujos de eventos

Crear etiqueta

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nueva etiqueta. Acto seguido deberá rellenar un formulario que contiene los datos de la etiqueta a crear, tales como el nombre de la etiqueta, la descripción, unidad de medida, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos

que se han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que registre la nueva etiqueta en la base de datos acabando aquí el caso de uso.

- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar, se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de etiqueta.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de la etiqueta pulsando el botón cancelar, terminando de esta forma el caso de uso.

Modificar etiqueta

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de una etiqueta se entra al detalle de la misma con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos de la etiqueta que se puedan modificar, tales como el nombre de la etiqueta, la descripción, la unidad de medida, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que ha introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registren los cambios de la etiqueta editada en la base de datos, acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla.

Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle de la etiqueta.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de una etiqueta pulsando el botón cancelar, terminando de esta forma el caso de uso.

Eliminar etiqueta

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de etiquetas se selecciona alguna o varias de ellas y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación se cerrará el diálogo sin llegar a eliminar nada. En caso contrario, al pulsar el botón de continuar se borran del sistema las etiquetas seleccionados.

2.1.7. Diagrama de gestión de tutoriales

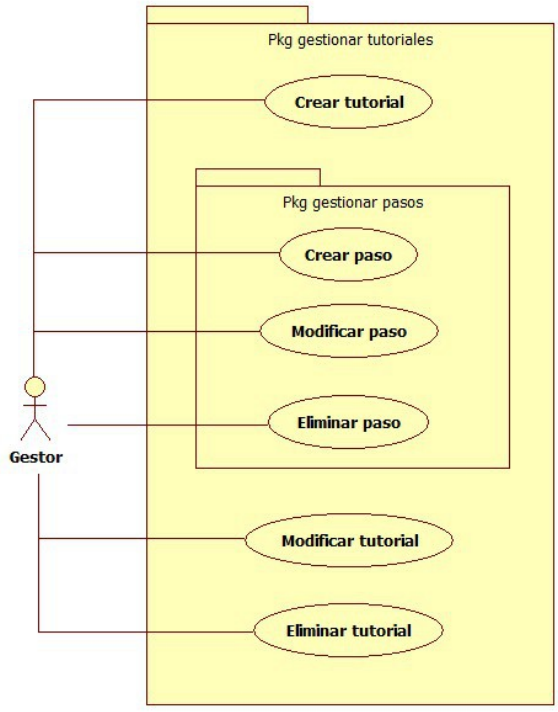


Ilustración 11: Casos de uso - Diagrama de gestión de tutoriales

Flujos de eventos

Crear Tutorial

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nuevo tutorial. Acto seguido deberá rellenar un formulario que contiene los datos del tutorial a crear, tales como el nombre del tutorial, la descripción, foto, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que se han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registre el nuevo tutorial en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar, se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de tutorial.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de tutorial pulsando el botón cancelar, terminando de esta forma el caso de uso.

Adicionalmente se podrá ejecutar el caso de uso de crear paso nuevo mediante un botón dedicado a esta función.

Modificar tutorial

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de un tutorial se entra al detalle del mismo con el botón de detalle/modificación.

En este momento se presenta un formulario con todos los datos del tutorial que se puedan modificar, tales como el nombre, la descripción, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá dos opciones:

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que se han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registren los cambios del tutorial editado en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle del tutorial.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de un tutorial pulsando el botón cancelar, terminando de esta forma el caso de uso.

Adicionalmente se podrán ejecutar los casos de uso caso de crear, modificar y eliminar un paso mediante botones dedicados a estas funciones.

Eliminar tutorial

Flujo de eventos principal:

El caso de uso empieza cuando después de realizar la búsqueda de tutoriales se selecciona alguno o varios de ellos y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación se cerrará el diálogo sin llegar a eliminar nada. En caso contrario al pulsar el botón de continuar se borrarán del sistema los tutoriales seleccionados.

Crear paso

Flujo de eventos principal:

El caso de uso empieza cuando el administrador/manager pulsa el botón de crear nuevo paso desde la pantalla de creación o modificación de tutoriales. Acto seguido deberá rellenar un formulario que contiene los datos del paso a crear, tales como el nombre del paso, la descripción, foto, etc.

Una vez el administrador/manager haya rellenado el formulario tendrá tres opciones:

- Pulsar el botón de asociar componentes. En este momento se navegará a la

pantalla de búsqueda de componentes en modo de selección. La pantalla de búsqueda tendrá multiselección en la tabla de resultados. Con el botón de aceptar volveremos a la pantalla anterior donde podremos visualizar los componentes asociados al paso.

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registre el nuevo paso en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar, se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de alta volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de alta de tutorial.

Flujo de eventos excepcional:

El administrador/manager puede cancelar un alta de paso pulsando el botón cancelar, terminando de esta forma el caso de uso.

El administrador/manager puede cancelar la asociación de componentes pulsando el botón de cancelar volviendo de esta forma a la pantalla de creación de paso sin haber asociado ningún componente.

Modificar paso

Flujo de eventos principal:

El caso de uso empieza cuando se selecciona uno de los pasos del tutorial y se pulsa el botón de modificar.

En este momento se presenta un formulario con todos los datos del paso que se puedan modificar, tales como el nombre, la descripción, la unidad de medida, etc.

Una vez el administrador/manager haya terminado de modificar el formulario tendrá tres opciones:

- Pulsar el botón de asociar componentes. En este momento se navegará a la pantalla de búsqueda de componentes en modo de selección. La pantalla de búsqueda tendrá multiselección en la tabla de resultados. Con el botón de aceptar volveremos a la pantalla anterior donde podremos visualizar los componentes asociados al paso.

- Pulsar el botón de guardar. En este momento el sistema validará que los datos que se han introducido en el formulario son correctos. En caso de ser incorrectos se mostrarán en el formulario los errores cometidos al cumplimentarlo. En caso contrario se enviará la información al servidor para que se registren los cambios del paso editado en la base de datos acabando aquí el caso de uso.
- Pulsar el botón de cancelar. En este momento se comprobará si el administrador/manager ya había modificado el formulario. En este caso se mostrará un mensaje de confirmación indicando que de continuar se perderán los datos introducidos.

  Si no se han realizado cambios en el formulario o aún así se decide continuar se cerrará el formulario de modificación volviendo al estado anterior de la pantalla.

  Si se han realizado cambios y se decide no continuar con la cancelación se cerrará el diálogo de confirmación de cancelación pudiendo así seguir modificando el formulario de detalle del paso.

Flujo de eventos excepcional:

El administrador/manager puede cancelar las modificaciones de un paso pulsando el botón cancelar, terminando de esta forma el caso de uso.

El administrador/manager puede cancelar la asociación de componentes pulsando el botón de cancelar volviendo de esta forma a la pantalla de creación de paso sin haber asociado ningún componente.

Eliminar paso

Flujo de eventos principal:

El caso de uso empieza cuando se selecciona uno de los pasos del tutorial y se pulsa el botón de eliminar. Se presentará un diálogo de confirmación para evitar eliminaciones accidentales. Si pulsamos el botón de cancelar la eliminación, se cerrará el diálogo sin llegar a eliminar nada. En caso contrario al pulsar el botón de continuar se borran del sistema los pasos seleccionados.

2.1.8. Diagrama de Internet

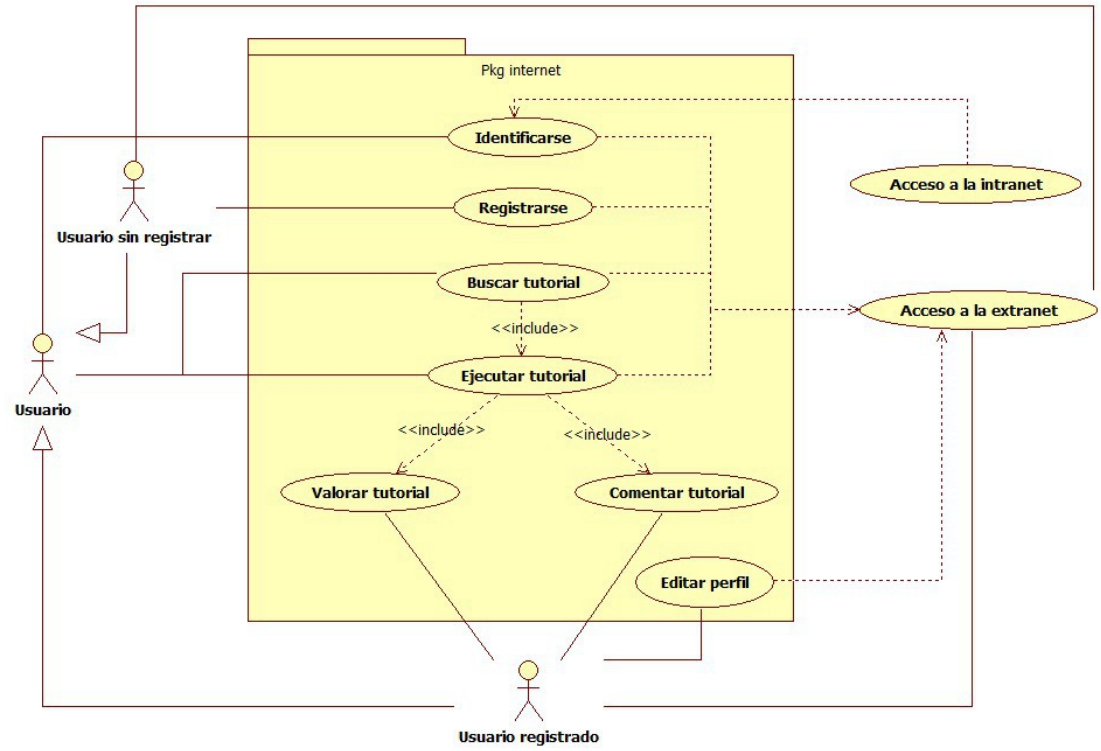


Ilustración 12: Casos de uso - Diagrama de internet

Flujos de eventos

Registrarse

Flujo de eventos principal:

El caso de uso comienza cuando un usuario que no tenga una sesión abierta en la aplicación pulsa en el botón "Registrarse". En este instante la aplicación mostrará una ventana con el formulario de alta de nuevos usuarios para que introduzca sus datos. Una vez el usuario haya guardado su usuario se mostrará un mensaje avisando de que ya puede loguearse en la aplicación.

Identificarse

Flujo de eventos principal:

El caso de uso comienza cuando un usuario que no tenga una sesión abierta en la aplicación rellena un formulario con su nombre de usuario y su contraseña y pulsa un

botón de “Login”. En este momento la aplicación comprobará si los datos introducidos existen en la base de datos. Si es así, abrirá una sesión en la aplicación para el usuario introducido. En caso contrario se le denegará el acceso.

La aplicación permitirá la entrada de tres tipos diferentes de usuarios registrados.

- Usuario plano:

  Este usuario no posee permisos especiales, por lo que no podrá acceder al módulo de administración de la aplicación. En la práctica la mayoría de los usuarios registrados tendrán este rol.

- Administrador:

  Este usuario no tendrá ninguna restricción dentro de la aplicación. Este rol está designado a aquellos usuarios que tengan que llevar la parte administrativa de la aplicación, configuración, y gestión de los distintos usuarios. En la práctica lo más probable es que como mucho haya dos o tres administradores.

- Manager:

  Este usuario está orientado a la gestión de los contenidos, pudiendo crear y modificar nuevos tutoriales así como realizar las labores de moderador censurando aquellos comentarios de usuarios que incumplan las normas de la aplicación.

Editar perfil

Flujo de eventos principal:

Los usuarios registrados tendrán la opción de ver y editar su perfil en una pantalla destinada a tal efecto. Todos los datos introducidos por el usuario serán editables salvo el nombre de usuario que le identifica en la aplicación.

Buscar tutorial

Flujo de eventos principal:

El caso de uso empieza cuando un usuario introduce sus términos de búsqueda y pulsa un botón de “Buscar”. En este momento la aplicación deberá mostrar los tutoriales que considere relevantes para el texto de búsqueda introducido por el usuario.

La búsqueda se presentará de dos formas diferentes, una búsqueda simple para los usuarios que deseen únicamente buscar por texto y una búsqueda avanzada para encontrar tutoriales que tengan o que no tengan ciertos componentes, que no tengan ciertas etiquetas entre sus componentes, etc.

Los resultados se podrán ordenar por relevancia, por dificultad, nota media de usuarios, etc.

En caso de no filtrar por ningún criterio en la búsqueda se mostrarán ciertos tutoriales relacionados con las valoraciones que ya haya hecho el usuario si está registrado o aquellos que más visitas o mejores valoraciones tengan si el usuario no está logueado en la aplicación.

Ejecutar tutorial

Flujo de eventos principal:

Al lanzar un tutorial se irán mostrando los diferentes pasos que lo conforman de uno en uno formando una secuencia que al llegar al último paso finalizará el tutorial ejecutado.

La forma de presentación de los pasos durante la ejecución será de forma ordenada, a pantalla completa y en letras grandes para que pueda seguirse, por ejemplo, a distancia mientras se realiza realmente lo que en ellos se muestra.

Cuando la ejecución de un tutorial finalice, automáticamente se mostrará una ventana para valorar y comentar el tutorial realizado.

Valorar tutorial

Flujo de eventos principal:

El flujo de eventos comienza cuando un usuario registrado rellena el campo de valoración del tutorial. En este momento se registrará dicha valoración sin requerirse ninguna otra acción.

La valoración se compondrá de una nota que el usuario atribuya al tutorial y la dificultad que ha tenido durante la realización del mismo.

Comentar tutorial

Flujo de eventos principal:

El flujo de eventos comienza cuando un usuario registrado rellena el formulario de comentario del tutorial y pulsa en el botón “Comentar”.

2.2. Modelo de datos

Los tutoriales están formados por un conjunto de pasos. Estos pasos normalmente requieren de un conjunto de componentes y/o útiles para llevarlo a cabo, p.ej. “atornillamos la escuadra de 10cm al tablero de 50x15x3cm con los dos tornillos din 933”.

Cada componente a su vez está asociado a un conjunto de etiquetas con un valor propio para él, p.ej el componente “pan de molde” tendrá las etiquetas valor energético, gluten.

Así la gente podría filtrar por ejemplo recetas que contengan un valor energético determinado o que no tengan gluten, por ejemplo, para los celíacos.

No se realizarán traducciones de los contenidos. En su lugar se crearán contenidos distintos en idiomas diferentes de la misma que lo hacen otros sitios Web como Wikipedia por ejemplo.

El modelo del dominio de la aplicación sería el siguiente:

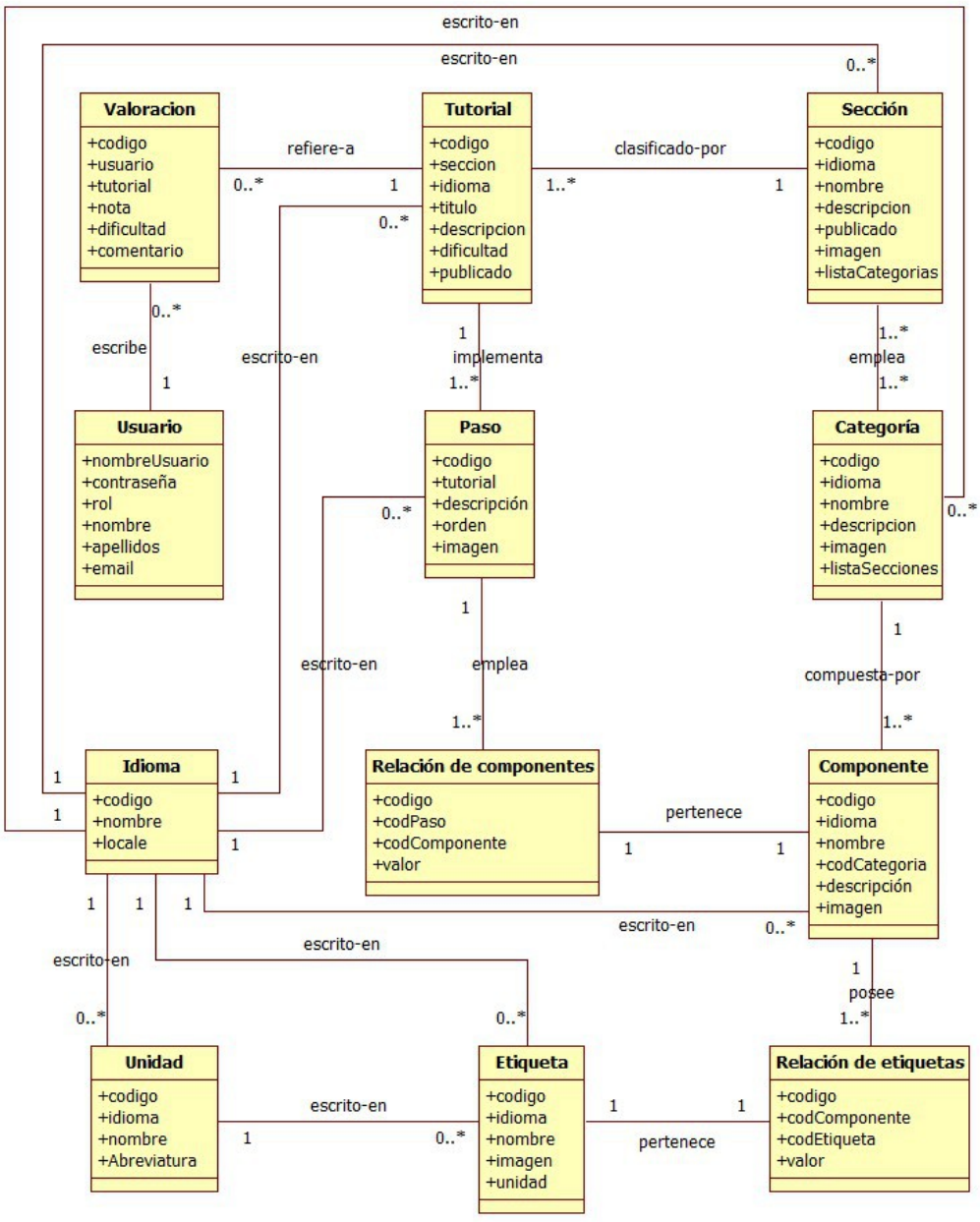


Ilustración 13: Modelo de datos

2.3. Requisitos no funcionales

Resumen:	**Aplicación Web modular y escalable**
Descripción:	Se requiere la implementación de una aplicación web modular y fácilmente escalable que permita la inclusión de nuevas funcionalidades de manera sencilla.

Tabla 16: Req. no funcionales - Aplicación Web modular y escalable

Resumen:	**La aplicación se dividirá en dos partes.**
Descripción:	La aplicación deberá dividirse en dos módulos, uno dedicado a los usuarios desde el cual se podrá navegar por los distintos tutoriales, ejecutarlos, etc. y otro para la parte administrativa tanto de la aplicación como de los contenidos.

Tabla 17: Req. no funcionales - La aplicación se dividirá en dos partes

Resumen:	**Navegadores soportados.**
Descripción:	Los clientes accederán a la interfaz de la aplicación a través del navegador web de su ordenador. Dada la gran cantidad de navegadores web que existen en el mercado es imposible llevar un mantenimiento de la aplicación que la haga usable para las distintas versiones de todos ellos, así que nos centraremos en los más habituales. Los navegadores web soportados por la aplicación serán Internet Explorer 9 o superior, Firefox, Chrome, opera y Safari.

Tabla 18: Req. no funcionales - Navegadores soportados

Resumen:	**Uso de cookies.**
Descripción:	La aplicación deberá hacer uso de cookies y alertar al usuario de ello como es la política actual de los sitios web. De esta forma en el logueo del usuario se dispondrá de un seleccionable para indicar que desea que se le recuerde en la aplicación y así no tener que iniciar sesión cada vez que ésta caduque. Así también, la posición dentro de la ejecución de un tutorial podrá guardarse para que la próxima vez que el usuario lo ejecute le pregunte si desea continuar donde lo dejó o por el contrario prefiere empezar desde cero.

Tabla 19: Req. no funcionales - Uso de cookies

3. DISEÑO

3.1. Arquitectura

Este sub-apartado contiene el diseño elaborado para el proyecto, el cual es producto de un análisis minucioso de los requisitos del sistema, según estos pueden ser satisfechos con las tecnologías y características elegidas.

Este sub-apartado está organizado alrededor de tres ideas principales:

1. Las características generales del diseño.
2. Los requisitos atendidos por el diseño.
3. Los modelos y vistas que lo detallan.

Al contrario que muchas otras actividades técnicas, el desarrollo de sistemas intensivos en software dedica la mayoría de sus esfuerzos a la especificación y modelado.

Los modelos son utilizados para el análisis de requisitos, el diseño de la solución y la especificación, construcción y despliegue del sistema en su ambiente de explotación.

Los modelos son presentados por medio de vistas o diagramas, generalmente utilizando notaciones gráficas como el UML.

Por otro lado, los programas de computadora son construidos por medio del uso de herramientas de traducción automáticas llamadas compiladores, para los cuales es construida la forma final y más detallada del software del sistema: el código fuente.

La ultima sección del sub-apartado indica la forma en que se puede obtener el código fuente del proyecto, así como las instrucciones de compilación necesarias para lograr la ejecución de los componentes que este código detalla.

3.1.1. Descripción general

Se entiende por arquitectura del software el conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente así como al comportamiento del software en tiempo de ejecución.

Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto. Esta sección describe en términos generales, las ideas principales detrás de la arquitectura escogida para el mismo.

3.1.2. Posicionamiento y alcance

Se trata de la construcción de una aplicación web en J2EE, que corra en un servidor web

con acceso desde Internet puesto que cualquier usuario debe poder conectar con la aplicación mediante el uso de un navegador web moderno a través de la URL de la aplicación.

Se dispondrá únicamente de un único computador para albergar todo el sistema de información en primera instancia, por lo que tanto el servidor de aplicaciones web empleado como el sistema elegido para la persistencia de datos convivirán dentro del mismo sistema operativo.

Estos parámetros definen el alcance inicial. Si en un futuro la aplicación sufriera un alto nivel de estrés, podría ser montada en un sistema distribuido.

La aplicación deberá contemplar un sistema de configuración simple y flexible que permita una cómoda implantación.

3.1.3. El albergue de la aplicación

El hardware y software disponible en la máquina donde se albergará la aplicación web y la cual puede influir en el diseño de la misma son los siguientes.

La máquina elegida para el albergue de los distintos sistemas sobre la que se sustentará la aplicación es un computador "Intel(R) Core(TM)2 Quad CPU Q6600 a 2.40GHz" a 64bits con 2GB de memoria RAM.

Dispone de un disco duro de 2TB formateado con ext4 donde corre la distribución de Linux Ubuntu 13.04.

La conexión a Internet es proporcionada por el ISP de Movistar con una conexión contratada de banda ancha ADSL de 10mbps.

El servidor de aplicaciones J2EE es un JBOSS 7.1 y dispone de un servidor de base de datos MYSQL 5.5.

Los accesos a trazas se realizarán mediante la consola de Linux de forma remota a través del servidor de SSH instalado en la máquina.

3.1.4. Características principales

El Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

Para ello MVC por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

El principio fundamental del patrón MVC es definir una arquitectura con claras responsabilidades para diferenciar componentes. En MVC hay tres participantes:

- Modelo:

  Un modelo representa los datos o reglas de negocio y el estado de la aplicación.

- Vista:

  Presenta los datos al usuario en un formato especificado.

- Controlador:

  El controlador maneja las solicitudes de las acciones realizadas por el usuario en la vista, actualiza el modelo y dirige a los usuarios la vista apropiada basándose en el resultado de la ejecución.

Este patrón de diseño se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

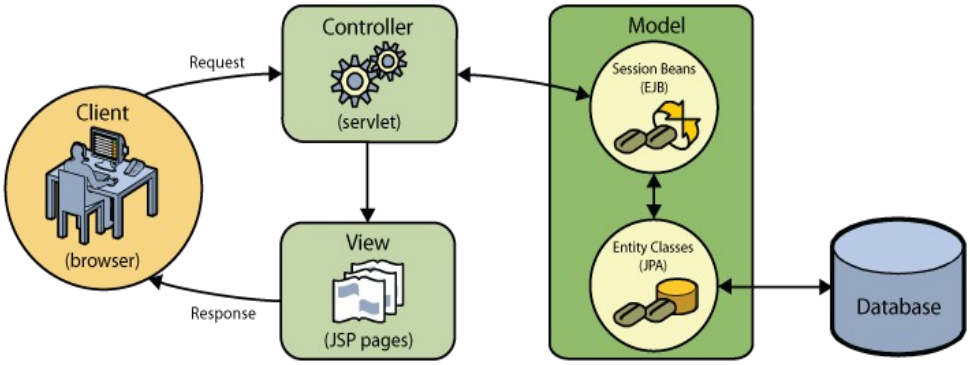


Ilustración 14: Modelo Vista Controlador

Es por eso que se utilizará este patrón de diseño para la construcción de la aplicación.

Aunque originalmente MVC fue desarrollado para aplicaciones de escritorio, ha sido ampliamente adaptado como arquitectura para diseñar e implementar aplicaciones web en los principales lenguajes de programación.

Se han desarrollado multitud de Frameworks, comerciales y no comerciales, que implementan este patrón. Estos Frameworks se diferencian básicamente en la interpretación de cómo las funciones MVC se dividen entre las de cliente y servidor.

Uno de estos Frameworks será el que se emplee en la construcción de la aplicación web.

3.1.4.1. Framework de Spring

En este apartado se describen el Framework seleccionado para la implementación de la aplicación.

3.1.4.1.1. Introducción al patrón Modelo-Vista-Controlador con Spring MVC

Para entender mejor el funcionamiento de Spring MVC será conveniente tener un claro conocimiento del patrón de diseño Modelo-Vista-Controlador explicado anteriormente.

Debido al auge que tienen las aplicaciones web basadas en Ajax, la aplicación del patrón MVC ha sido mejorada para proveer mejores experiencias a los usuarios usando JavaScript, Ajax y el uso de formatos especiales de información como son JSON o XML.

En la siguiente imagen se muestra el concepto de mejora en el patrón MVC.

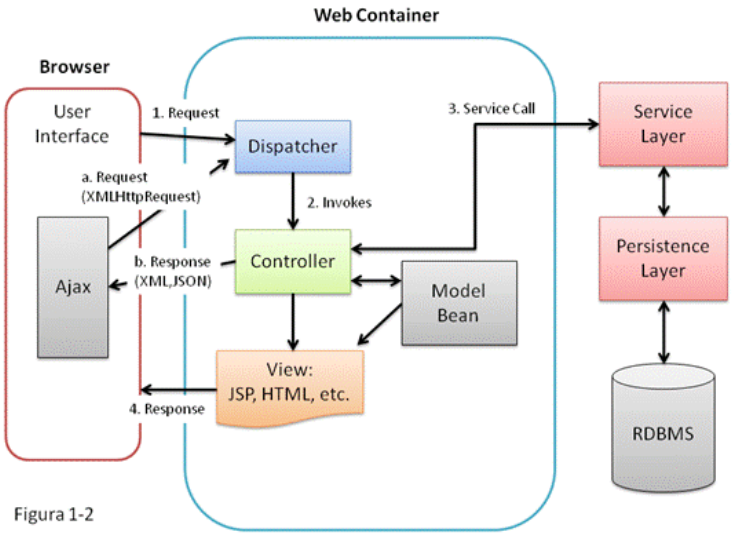


Ilustración 15: Modelo Vista Controlador Mejorado

Esta figura ilustra un patrón de aplicaciones web usado comúnmente, el cual puede ser tratado como la mejora del tradicional patrón MVC.

1. Request:

   Una petición es enviada al servidor, la mayoría de los Frameworks (Spring MVC, Struts etc) tendrán un dispatcher para atender las solicitudes.

2. Invokes:

   El dispatcher envía las solicitudes al controlador apropiado.

3. ServiceCall:

   El controlador interactúa con la capa de servicio para hacer uso de la capa de persistencia.

4. Response:

   El controlador actualiza el modelo basado en el resultado y responde con la

correspondiente vista al usuario.

En adición con las llamadas Ajax sucede lo siguiente:

a. Request: Un XMLHttpRequest (Petición Ajax) es preparado y enviado al servidor. El dispatcher enviará la solicitud al correspondiente controlador.
b. Response: El controlador interactúa con la capa de servicio. Los datos de respuesta serán formateados y enviados al navegador. En este caso las vistas no están involucradas ya que el navegador recibe los datos y realiza una actualización parcial en la vista existente.

3.1.4.1.2. Introducción a Spring MVC

Spring MVC es uno de los módulos del Framework Spring, y éste provee un exhaustivo soporte para el patrón MVC. Además también provee soporte de otras características, una de ellas es facilitar la implementación de la capa de presentación.

3.1.5. Restricciones y limitaciones

Las limitaciones actuales en el proyecto vienen dadas por la única máquina disponible para albergar la aplicación y el hecho de que se cuenta con un presupuesto inicial de 0€, lo cual el software a emplear viene limitado por licencias gratuitas siendo preferidas entre ellas las de licencia “open source”.

También habrá que tener en cuenta las limitaciones temporales en cuanto a las soluciones que serán implementadas.

3.1.5.1. Vistas

Los sistemas intensivos en software se encuentran formados por un conjunto de componentes que son los elementos listos para ser ejecutados, producidos por el proyecto.

Dichos componentes se distribuyen sobre los distintos equipos según lo que se detalla en la vista de despliegue.

3.1.5.1.1. Vista de componentes

El diagrama de componentes nos muestra las capas en las que se divide el sistema. En este caso son Vista, Controlador y Modelo.

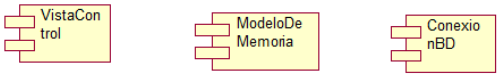


Ilustración 16: Vista de componentes

3.1.5.1.2. Vista de despliegue

El despliegue es la planificación de qué componentes se ejecutarán en los distintos computadores (nodos) relacionados con el sistema.

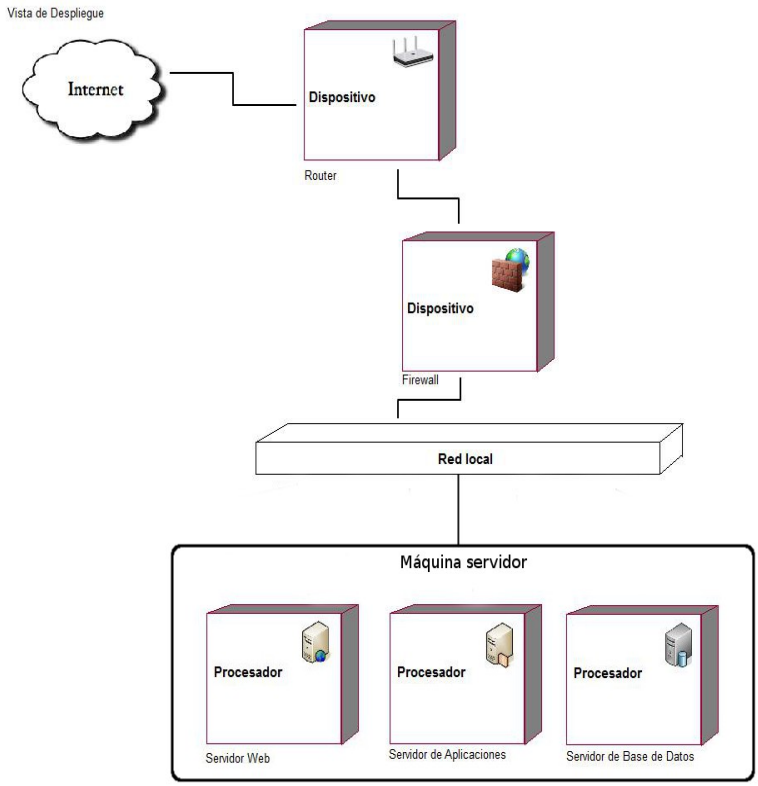


Ilustración 17: Vista de despliegue

3.2. Diseño de la base de datos

El diseño de una base de datos consiste en definir la estructura de los datos que debe tener la base de datos del proyecto. Dado que el sistema de gestión de base de datos (SGBD) seleccionado es relacional (MySQL), la estructura de datos será un conjunto de esquemas de relación con sus atributos, dominios de atributos, claves primarias, claves foráneas, etc.

Basándome en el análisis realizado en el modelo de dominio del proyecto es hora de ampliar el diseño conceptual de la base de datos a un modelo de entidad-relación (modelo ER) listo para servir de guión en la implementación de los scripts (guiones) de la creación de base de datos en MySQL:

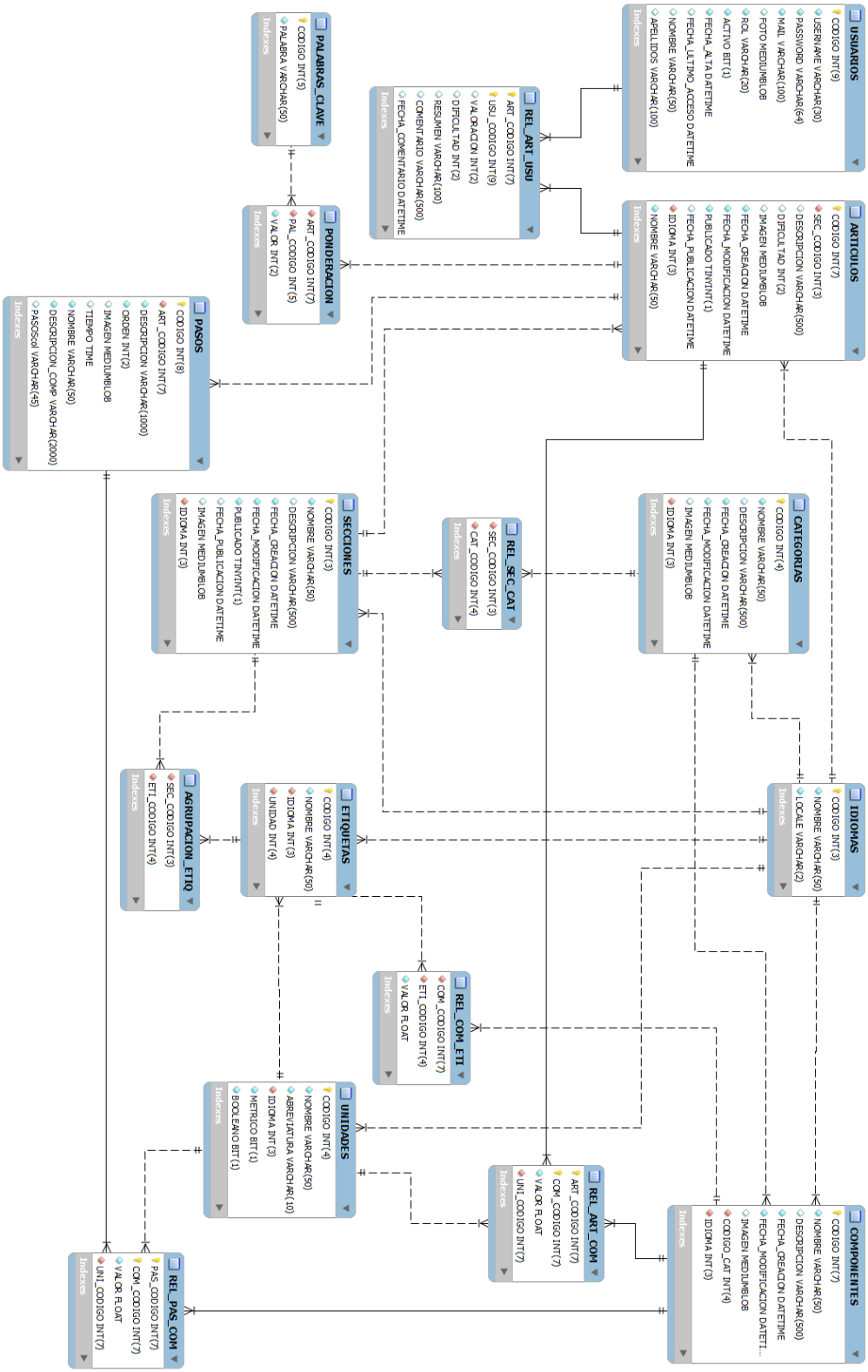


Ilustración 18: DiagramaER

3.3. Presentación

En este apartado se presentan las decisiones tomadas respecto de cómo se visualizará la aplicación, tecnologías empleadas, etc.

3.3.1. Tipo de interfaz

En la ingeniería de software se denomina aplicación Web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor Web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web a la que se confía la ejecución de la interfaz al navegador.

Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo y a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales.

Es importante mencionar que una página Web puede contener elementos que permitan una comunicación activa entre el usuario y la información. Esto posibilita que el usuario acceda a los datos de modo interactivo gracias a que la página responderá a cada una de sus acciones. Por ejemplo rellenar y enviar formularios que modifiquen la base de datos sobre la que se nutre la propia aplicación.

La aplicación Web deberá ser visible a través de las últimas versiones de los navegadores:

- Chrome
- Mozilla Firefox
- Internet Explorer
- Opera
- Safari

3.3.2. Interfaz responsiva o sensible

La interfaz deberá ser responsiva, adaptándose tanto a resoluciones grandes como a pequeñas. La resolución mínima de la aplicación será de 600 píxeles de ancho, abarcando así aquellos dispositivos móviles de gama baja o que lleven ya bastante tiempo en el mercado.

El uso de dispositivos móviles está creciendo a un ritmo increíble, tablets y smartphones han incrementado sus ventas en los últimos años y la navegación en Internet mediante estos dispositivos es cada vez más común. Ese es el motivo por el que el diseño web

adaptativo se ha vuelto tan popular. Es una técnica que proporciona una solución Web que puede manejar la visualización de escritorio y dispositivos móviles.

Con una sola versión en HTML y CSS se cubren todas las resoluciones de pantalla, es decir, el sitio web creado estará optimizado para todo tipo de dispositivos (PCs, tabletas, teléfonos móviles...). Esto mejora la experiencia del usuario, a diferencia de lo que ocurre, por ejemplo, con sitios web de ancho fijo cuando se acceden desde dispositivos móviles.

De esta forma se reducen los costes de creación y mantenimiento, pues se evita tener que desarrollar aplicaciones ad-hoc para versiones móviles. Teniendo que crear por ejemplo una aplicación específica para iPhone, otra para móviles Android, etc.

Twitter Bootstrap:

Para la consecución de una interfaz responsiva se decide el uso de Twitter Bootstrap.

Twitter Bootstrap es una colección de herramientas de software libre para la creación de sitios y aplicaciones web. Bootstrap fue desarrollado por los creadores de Twitter como un marco de trabajo (Framework) para fomentar la consistencia a través de herramientas internas.

Es una hoja de estilos que lleva el antiguo concepto de maquetación con tablas a los tiempos actuales mediante capas que cargan el rol de las celdas.

Se empleará la última versión disponible, la 3.0. Desde la versión 2.0 se comenzó el soporte a diseños sensibles. Esto significa que el diseño gráfico de la página se ajusta dinámicamente, tomando en cuenta las características del dispositivo usado.

Bootstrap es compatible con la mayoría de navegadores web.

3.3.3. JavaScript

Las aplicaciones web generan dinámicamente una serie de páginas en un formato estándar, como HTML o XHTML, soportados por los navegadores web comunes. Se utilizan lenguajes interpretados en el lado del cliente, directamente o a través de plugins tales como JavaScript para añadir elementos dinámicos a la interfaz de usuario. Generalmente cada página web se envía al cliente como un documento estático, pero la secuencia de páginas ofrece al usuario una experiencia interactiva.

Es por esto que parte de la presentación HTML será generada en el servidor y presentada al navegador del cliente (la parte estática que se presentará al usuario). Otra gran parte recaerá en el propio navegador a través de código JavaScript que deberá interpretar y ejecutar.

Por ser una de las tecnologías más punteras de JavaScript y tener en cuenta un abanico amplio de navegadores se opta por el uso de jQuery en su última versión hasta la fecha (1.10).

jQuery:

jQuery es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM (API de objetos que conforman los documentos HTML), manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. jQuery es en la actualidad la biblioteca de JavaScript más utilizada.

Una de las principales razones por las que es recomendado su uso es porque ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían mucho más código. Es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Con esta librería de JavaScript cubriremos las dependencias de otras librerías y plugins que empaquetan funcionalidades extra que serán de gran utilidad y que se enumeran a continuación.

- jQuery UI:

  jQuery UI es una biblioteca de componentes para el Framework jQuery que le añade un conjunto de plugins, widgets y efectos visuales para la creación de aplicaciones web. Cada componente o módulo se desarrolla de acuerdo a la filosofía de jQuery.

  En la página oficial de jQueryUI existe una herramienta para crear una versión personalizada de la biblioteca para que sólo se incluyan los módulos que se van a utilizar. De esta manera se optimiza el tamaño del fichero, conteniendo sólo lo necesario.

- Twitter Bootstrap:

  Aparte de la hoja de estilos para hacer la interfaz responsiva, Bootstrap también posee un conjunto de plugins de jQuery orientados al uso de su hoja de estilos.

  Los componentes de JavaScript para Bootstrap están basados en la librería jQuery de JavaScript. Proveen elementos adicionales de interfaz de usuario como diálogos, tooltips y carruseles. También extienden la funcionalidad de algunos elementos de interfaz existentes incluyendo por ejemplo una función de autocompletar para campos de entrada (input). La versión 3.0 soporta los siguientes plugins de JavaScript:

  - Modal
  - Dropdown
  - Scrollspy
  - Tab
  - Tooltip

- Popover
- Alert
- Button
- Collapse
- Carousel
- Typeahead

Flash:

En ningún momento se incluirá contenido flash en la aplicación.

3.4. Documentos de diseño

En este apartado del diseño se encapsulan los distintos documentos de diseño creados para guiar la implementación de todos los módulos que componen la aplicación.

Estos documentos de diseño están enfocados a las distintas pantallas que componen la aplicación. Describen con riguroso detalle todos los elementos que componen la interfaz y su funcionamiento así como el plan de pruebas específico de la pantalla.

Una vez especificado el diseño se define una estimación temporal del tiempo de desarrollo para la pantalla.

Todos los documentos de diseño constarán de los mismos subapartados:

- Caso o casos de uso a implementar.
- Objetos del modelo y entidades de la base de datos.
- Implementación de la seguridad.
- Diseño de la pantalla.
- Diseño funcional.
- Control de excepciones.
- Plan de pruebas.
- Planificación.

3.4.1.1. Diseño de la cabecera

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla principal serán los referentes al diagrama de Internet donde se especifica en el análisis que un usuario puede:

- Identificarse.
- Registrarse.
- Editar perfil.

Las funciones implementadas por la cabecera serán comunes a todas las pantallas de todos los módulos de la aplicación. De esta forma tratamos la cabecera como un módulo aparte con sus propias funcionalidades.

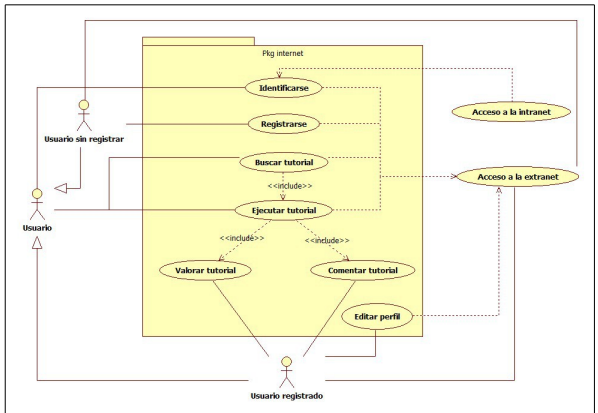


Ilustración 19: Diseño de la cabecera - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

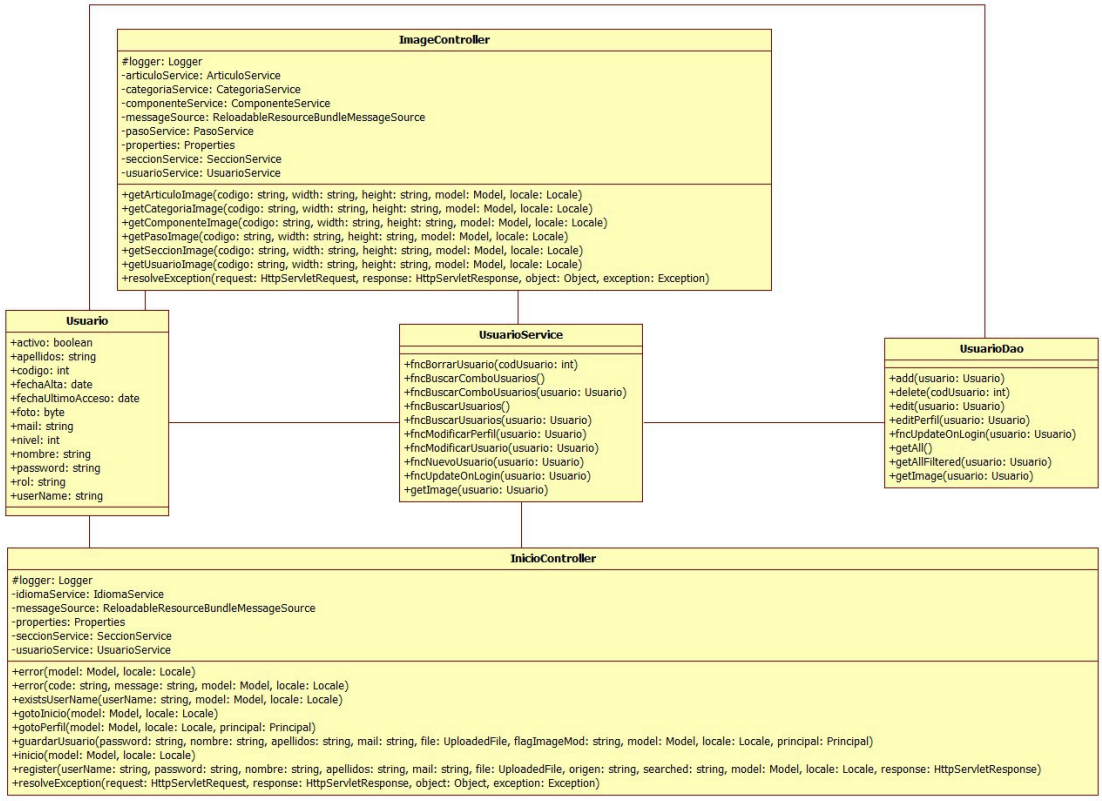


Ilustración 20: Diseño de la cabecera – Clases empleadas

Entidades empleadas:

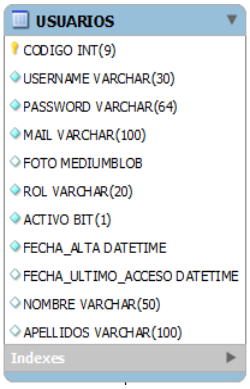


Ilustración 21: Diseño de la cabecera - Entidades empleadas

Implementación de la seguridad

Cualquier tipo de usuario podrá acceder a la pantalla principal. Las particularidades de lo que verán los distintos perfiles se especificarán en el diseño de la pantalla.

Diseño de la pantalla

En la parte izquierda de la cabecera se mostrará el título de la aplicación con un texto sobre el que se aplica el componente de FitText (http://fittextjs.com/). De esta forma el tamaño del título variará dependiendo del espacio destinado para él según las distintas resoluciones en las que se visualice la aplicación.

El título de la aplicación será un enlace que navegará a la pantalla principal de la misma.

Un usuario que no esté logueado en la aplicación visualizará en la cabecera los siguientes campos:

- Campo de texto para el nombre de usuario:

  El campo de texto no tendrá label, en su lugar mostrará un placeholder con el texto "Usuario". El input se formará con el componente "input group" de Bootstrap (http://getbootstrap.com/components/#input-groups). La imagen de cabecera de campo será el icono de glyphicons de Bootstrap "glyphicon-user" (http://getbootstrap.com/components/#glyphicons).

- Campo de texto para la contraseña:

  El campo de texto no tendrá label, en su lugar mostrará un placeholder con el texto "Contraseña". El input se formará con el componente "input group" de Bootstrap. La imagen de cabecera de campo será el icono de glyphicons de Bootstrap "glyphicon-asterisk".

- Botón para registrarse como nuevo usuario:

  Será un botón de tamaño normal por defecto de Bootstrap (http://getbootstrap.com/css/#buttons). El botón contendrá un span precediendo al texto con el glyphicon de Bootstrap "glyphicon-edit". Al hacer clic en el botón se abrirá una ventana modal con el componente "bootstrap3-dialog" (http://nakupanda.github.io/bootstrap3-dialog/) que mostrará un formulario con los siguientes datos:

  - Usuario:

    Campo de texto obligatorio.

  - Contraseña:

    Campo de texto de tipo password obligatorio.

  - Confirmación de contraseña:

Campo de texto de tipo password obligatorio que validará si lo escrito es exacto al campo anterior de contraseña.

- Nombre:

  Campo de texto obligatorio.

- Apellidos:

  Campo de texto obligatorio.

- E-mail:

  Campo de texto obligatorio que comprobará si es una dirección de e-mail válida.

- Imagen:

  Input de tipo file. No es obligatorio.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente y en caso satisfactorio lanzará la acción de registrar un nuevo usuario.

- Botón de cancelar:

  Será un botón de tamaño normal por defecto de Bootstrap. Cancela el registro cerrando el diálogo con el formulario de alta.

- Botón para loguearse en la aplicación:

  Será un botón de tamaño normal por defecto de Bootstrap. El botón contendrá un span precediendo al texto con el glyphicon de Bootstrap “glyphicon-log-in”. Este botón enviará el formulario por método POST a la dirección atendida por spring-segurity para el logueo (/j_spring_security_check).

Un usuario logueado en la aplicación (da igual su perfil en la misma) visualizará en la cabecera los siguientes campos:

- Su foto de perfil:

  La foto de perfil de usuario será un enlace que mostrará un diálogo con las opciones que puede realizar el usuario logueado. Éste diálogo se realizará mediante un “popover” de Bootstrap (http://getbootstrap.com/javascript/#popovers) y contendrá los siguientes enlaces:

  - Administrar:

    Este enlace vendrá precedido al texto por un span con el glyphicon de Bootstrap “glyphicon-cog”. Únicamente se mostrará para usuarios logueados con perfil de administrador o mánager. En caso de tratarse de un usuario con

perfil de administrador navegará a la pantalla de administración, mientras que de tratarse de un usuario con perfil de mánager irá a la pantalla de gestión de contenidos.

- Mi perfil:

  Este enlace vendrá precedido al texto por un span con el glyphicon de Bootstrap “glyphicon-user”. El enlace navegará a la pantalla de perfil del usuario logueado.

- Salir:

  Este enlace vendrá precedido al texto por un span con el glyphicon de Bootstrap “glyphicon-log-out”. El enlace navegará a la dirección atendida por spring-segurity para el deslogueo (/j_spring_security_logout).

Diseño funcional

Las funciones que se ejecutarán en el servidor desde la cabecera serán las siguientes:

- Log-in en la aplicación:

  Esta función viene implementada por spring-security puesto que usa un Handler de autentificación por defecto escuchando peticiones a la ruta “/j_spring_security_check”. En el siguiente diagrama de secuencia se puede observar el funcionamiento del mismo:

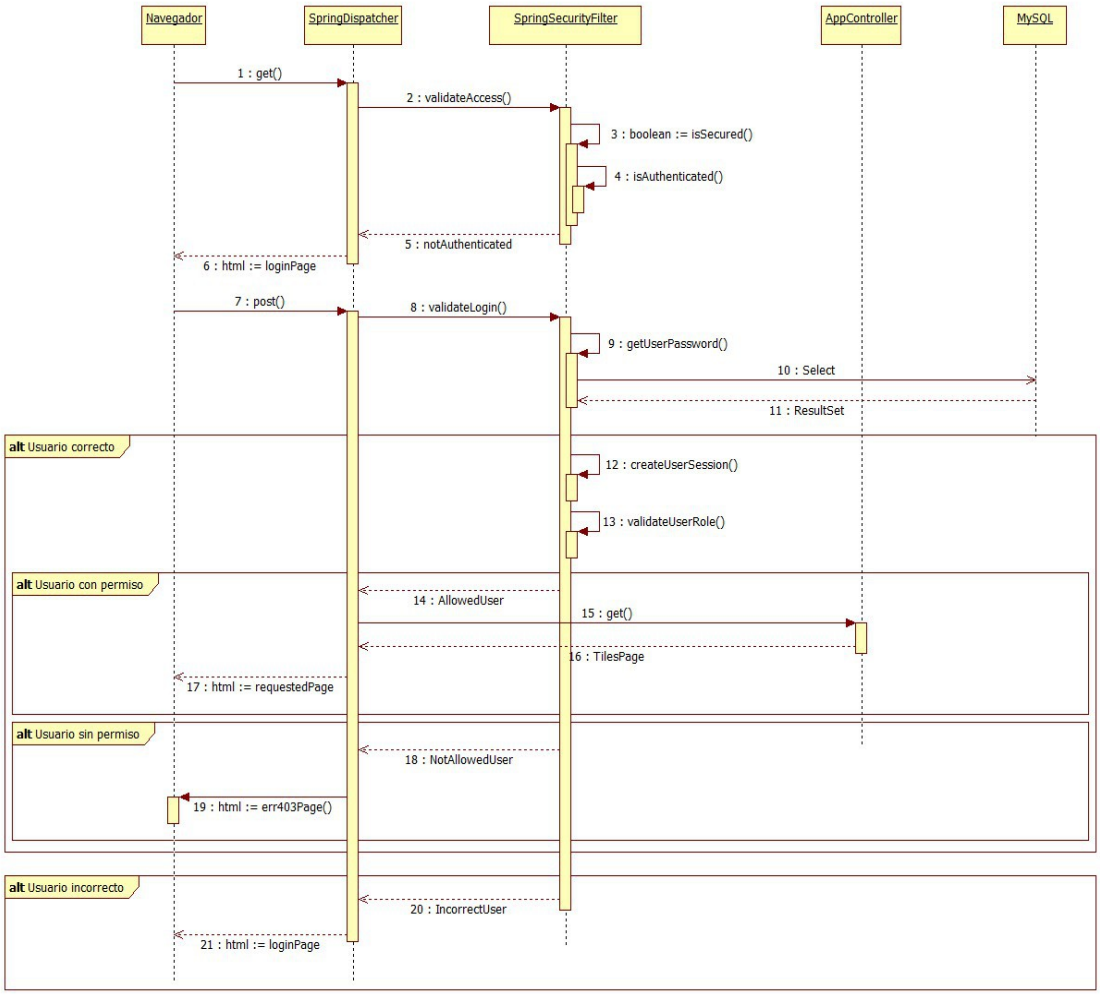


Ilustración 22: Diseño de la cabecera - Diagrama de secuencia de la seguridad

La petición post que realiza el login sería la correspondiente a la acción número 7 en el diagrama. Si el usuario es correcto Spring Security guardará sus datos en el contexto de Spring como usuario logueado y retornará a la pantalla que se estaba visualizando. En caso de un login erróneo esto no se producirá y se retornará a la misma pantalla sin resultado alguno.

- Log-out en la aplicación:

  Esta función viene implementada por Spring Security puesto que usa un Handler de autentificación por defecto escuchando peticiones a la ruta "/j_spring_security_logout". Spring Security eliminará los datos del usuario logueado del contexto de Spring y mostrará la página principal.

- Alta de nuevo usuario:

  Esta función está dedicada al alta de nuevos usuarios.

Alojada en la clase controlador de Spring InicioController su especificación será la siguiente:

```
@RequestMapping(value = "register", method = RequestMethod.POST)
   public String register(
           @ModelAttribute("userName") String userName
         , @ModelAttribute("password") String password
         , @ModelAttribute("nombre") String nombre
         , @ModelAttribute("apellidos") String apellidos
         , @ModelAttribute("mail") String mail
         , @ModelAttribute("FORM") UploadedFile file
         , @ModelAttribute("origen") String origen
         , @ModelAttribute("searched") String searched
         , Model model, Locale locale
         , HttpServletResponse response) {
   //TODO
}
```

Deberá verificar que los datos obtenidos de presentación son correctos y llamar al la función fncNuevoUsuario del servicio UsuarioServiceImpl para que guarde el usuario en la base de datos.

Una vez el usuario se haya dado de alta correctamente en la base de datos redirigiremos la navegación a una pantalla de feedback de usuario registrado. Ahí se mostrará: un mensaje al usuario informándole de que ya dispone de usuario válido y un enlace a la pantalla principal.

Control de excepciones

Las excepciones que se den a la hora de navegar a otras pantallas se controlarán internamente por los handlers implementados a tal efecto en la arquitectura de la aplicación.

Las excepciones que resulten en el registro de un nuevo usuario mostrarán un feedback informando del error en la pantalla de feedback de registro.

Plan de pruebas

- Deben satisfacerse todas las pruebas unitarias facilitadas de antemano.
- Registrar un usuario nuevo.
- Comprobar que no se permite el registro de no validarse correctamente el formulario de alta.
- Comprobar en la base de datos que los datos del usuario se han guardado correctamente.
- Comprobar que el mensaje de feedback de alta se muestra correctamente tras el alta.
- Loguearse con el usuario recién registrado.

- Comprobar que se permite el login en la aplicación y se poseen permisos de usuario registrado (sale la foto del usuario en la cabecera y no aparece el botón "Administrar" en el popover).

Planificación

Para la construcción de este módulo se calcula aproximadamente un 17% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (17% sobre 1000 horas aprox.)	59 horas
Consumidas en diseño	9 horas
Disponibles para implementación	50 horas

Tabla 20: Diseño de la cabecera - Planificación

3.4.1.2. Página principal

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla principal serán los referentes al diagrama de Internet donde en el análisis se especifica que un usuario puede:

- Buscar tutoriales

Esta pantalla es la primera página de la aplicación que se mostrará a todos los usuarios. Prácticamente no tiene funcionalidad salvo la de presentar una introducción al usuario y facilitarle el acceso a la información que esté buscando. Es por este motivo que el único caso de uso al que hace referencia es a la búsqueda de tutoriales; la función primaria de la aplicación.

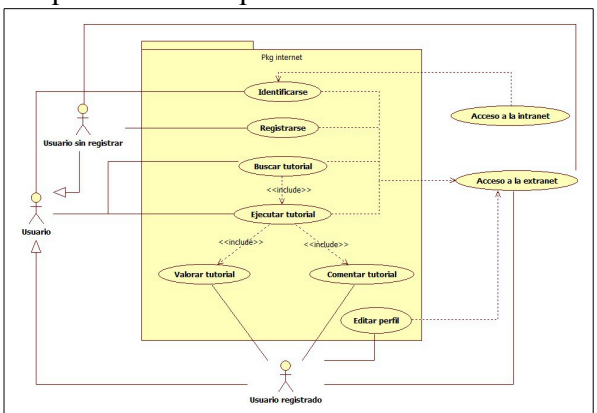


Ilustración 23: Página principal - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

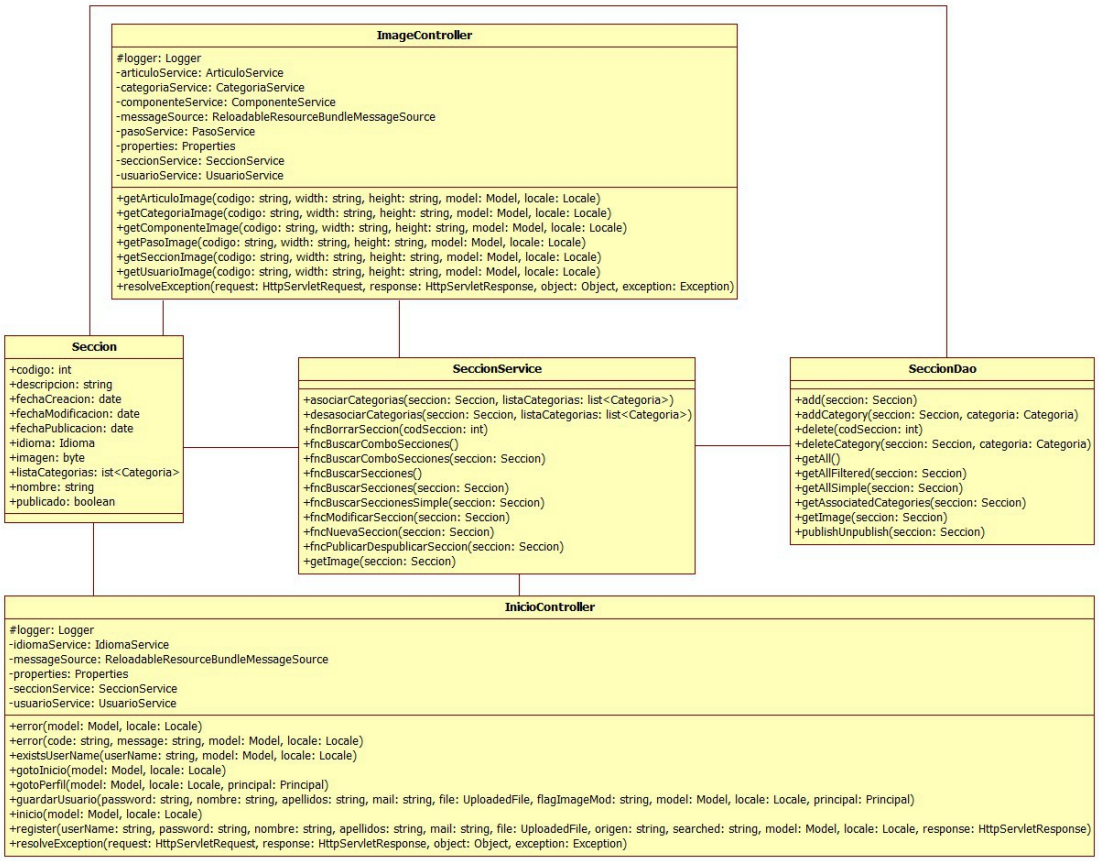


Ilustración 24: Página principal – Clases empleadas

Entidades empleadas:

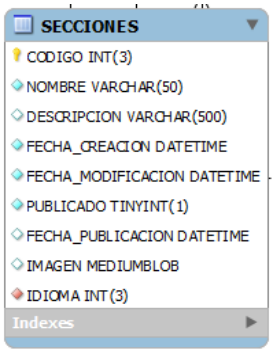


Ilustración 25: Página principal - Entidades empleadas

Implementación de la seguridad

Cualquier tipo de usuario podrá acceder a la pantalla principal. No se contemplan particularidades en la vista dependiendo de los perfiles de usuario.

Diseño de la pantalla

La pantalla debe ser llamativa y muy simple. Mostrará un mensaje de bienvenida a la aplicación y una descripción sobre el significado, importancia, etc. de los tutoriales en la vida cotidiana. También mostrará un slider o carrusel con las secciones temáticas de tutoriales disponibles.

La estructura de la sección del cuerpo quedará de la siguiente forma.

1. Artículo con la bienvenida a la aplicación.
2. Slider con las secciones disponibles.
3. Artículo con información genérica sobre los tutoriales.

El slider se implementará con el componente SlitSlider (http://tympanus.net/codrops/2012/06/05/fullscreen-slit-slider-with-jquery-and-css3/) y será sensible a eventos táctiles de deslizamiento a izquierda y derecha entre elementos del carrusel empleando el componente TouchSwype (http://labs.rampinteractive.co.uk/touchSwipe).

Diseño funcional

Las funciones que se ejecutarán en el servidor para la pantalla principal serán las siguientes:

- Búsqueda de secciones publicadas:

  Esta función deberá ser llamada antes de presentarse la pantalla principal.

  Alojada en la clase controlador de Spring InicioController su especificación será la siguiente:

```
@RequestMapping(value = "/", method = RequestMethod.GET)
   public String inicio(Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncBuscarSecciones del servicio SeccionService para obtener los datos de las secciones que estén publicadas en la base de datos. Una vez metida la lista de secciones en el modelo de Spring redireccionará a la vista de la pantalla principal.

- Búsqueda de tutoriales de una sección:

  Esta función deberá ser llamada cuando se haga clic sobre una de las secciones del slider. Alojada en la clase controlador de Spring ArticuloController su especificación será la siguiente:

```
@RequestMapping(
           value="/**"
         , method = RequestMethod.GET)
   public String buscarArticulos(
           @ModelAttribute("filtro") Articulo pFiltro
         , Model model
         , Locale locale
         , Principal principal) {
   //TODO
}
```

  Esta función deberá llamar a la función fncBuscarArticulosNombrePasos del servicio ArticuloService para obtener los datos de los artículos publicados disponibles para la sección seleccionada. Una vez metida la lista de artículos en el modelo de Spring redireccionará a la vista de la pantalla de presentación de tutoriales.

Control de excepciones

Las excepciones que se den a la hora de navegar a otras pantallas se controlarán internamente por los handlers implementados a tal efecto en la arquitectura de la aplicación.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Deben satisfacerse todas las pruebas unitarias facilitadas de antemano.
- Comprobar que las secciones mostradas son todas las secciones almacenadas en la base de datos con el campo PUBLICADO=1.
- Validar que los eventos del slider funcionan correctamente.
- Comprobar que navega correctamente a la pantalla de presentación de tutoriales y que muestra todos los tutoriales que cumplan PUBLICADO = 1 AND COD_SECCION = <codigo sección seleccionada>.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 3% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (3% sobre 1000 horas aprox.)	34 horas
Consumidas en diseño	3 horas
Disponibles para implementación	31 horas

Tabla 21: Página principal - Planificación

3.4.1.3. Presentación de tutoriales

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes al diagrama de Internet. En el análisis se especifica que un usuario puede:

- Buscar tutoriales
- Ejecutar tutoriales
- Valorar tutoriales
- Comentar tutoriales

Adicionalmente contemplaremos que cualquier usuario sea capaz de compartir cualquier tutorial en las redes sociales que se configuren dentro de la aplicación.

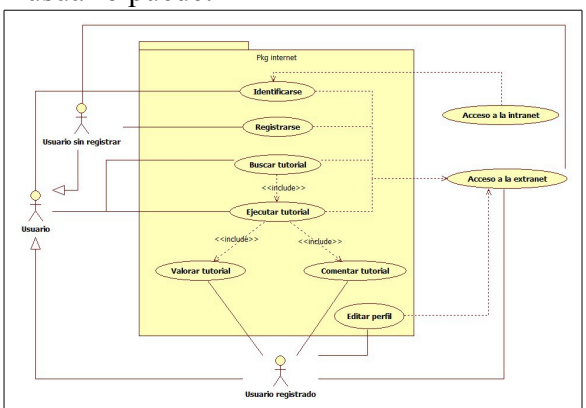


Ilustración 26: Presentación de tutoriales - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

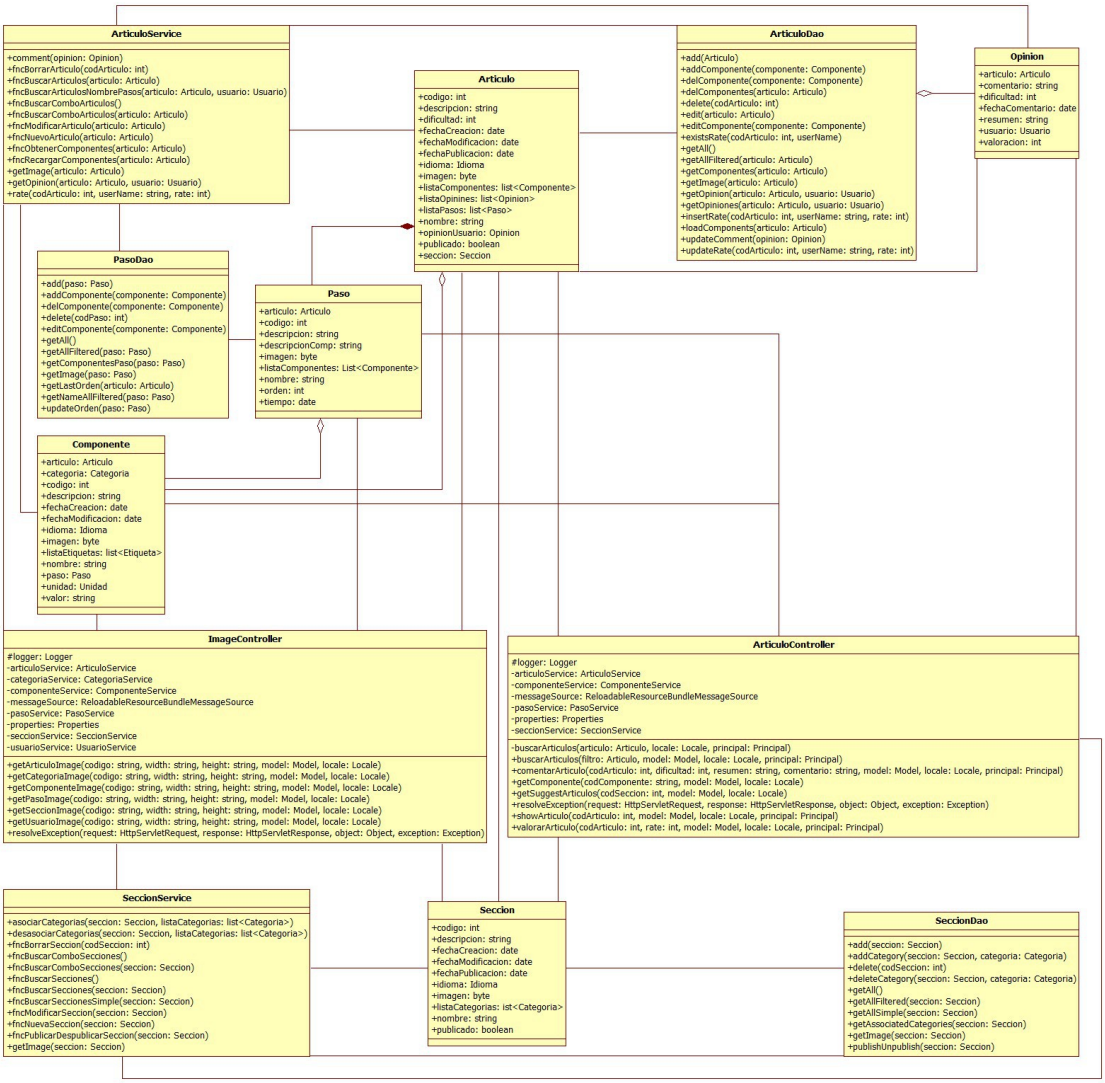


Ilustración 27: Presentación de tutoriales – Clases empleadas

Entidades empleadas:

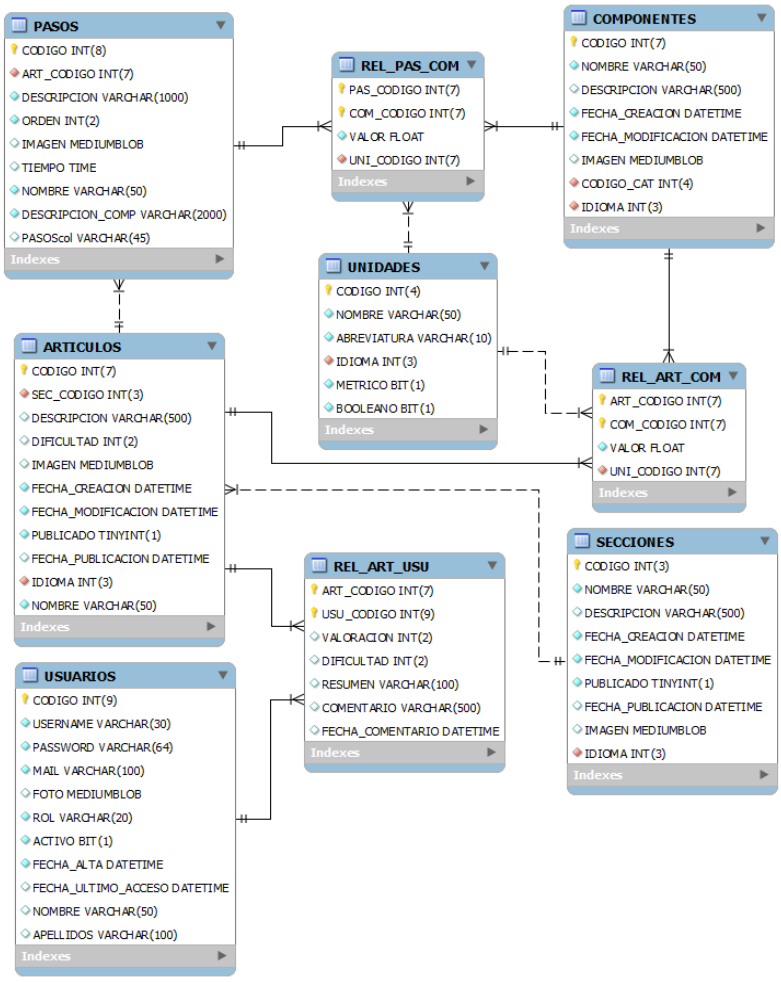


Ilustración 28: Presentación de tutoriales - Entidades empleadas

Implementación de la seguridad

Cualquier tipo de usuario podrá acceder a la pantalla principal. Las particularidades de lo que verán los distintos perfiles se especificarán en el diseño de la pantalla.

Diseño de la pantalla

El cuerpo de la pantalla presentará la sección con un primer artículo dedicado al formulario de búsqueda de tutoriales seguido otros tantos artículos como tutoriales se hayan encontrado en la búsqueda.

En el artículo para el filtrado se mostrará un fieldset que tendrá por leyenda “Criterios de búsqueda”. En su interior albergará un formulario con un combo con las secciones

publicadas en el idioma de la sesión del usuario y un campo de texto para el nombre del tutorial a buscar. El campo de búsqueda de nombre se formará con un input-group de Bootstrap que tendrá a la derecha un botón por defecto de Bootstrap sin texto y con el glyphicon "glyphicon-search".

Se requiere que este campo sugiera tutoriales que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI (http://jqueryui.com/autocomplete/)

Cada uno de los artículos de tutorial que se encuentren contendrán un panel por defecto de Bootstrap (http://getbootstrap.com/components/#panels). Este panel tendrá por cabecera el título del tutorial y las acciones que pueden realizarse. En el cuerpo del panel se dispondrá la imagen del tutorial a la izquierda y la descripción de éste a la derecha. Cabe destacar que al hacer clic sobre la imagen ésta se descargará y presentará a tamaño completo en pantalla usando el componente BlockUI (http://malsup.com/jquery/block/).

Las acciones presentadas en la cabecera del panel se encontrarán a la derecha representadas por botones por defecto de Bootstrap. Se describen a continuación:

- Resumen:

  Únicamente contiene el glyphicon "glyphicon-list".

  Al pasar por encima con el ratón se mostrará, con el componente tooltip de Bootstrap (http://getbootstrap.com/javascript/#tooltips), el mensaje "Resumen".

  Al hacer clic en el botón se desplegará un footer del panel con el resumen de los pasos a seguir en el tutorial.

- Lista de la compra:

  Únicamente contiene el glyphicon "glyphicon-shopping-cart".

  Al pasar por encima con el ratón se mostrará, con el componente tooltip de Bootstrap, el mensaje "Lista de la compra".

  Al hacer clic en el botón se desplegará un footer del panel con la lista de componentes y cantidades necesarias para la consecución del tutorial.

- Comentar:

  Este botón sólo se presentará a aquellos usuarios que estén logueados en la aplicación.

  Únicamente contiene el glyphicon "glyphicon-comment".

  Al pasar por encima con el ratón se mostrará, con el componente tooltip de Bootstrap, el mensaje "Comentar".

  Al hacer clic en el botón se desplegará un footer del panel con los campos para comentar y valorar el tutorial así como las valoraciones y comentarios que otros usuarios hayan publicado previamente.

Los campos del formulario de comentarios y valoración serán los siguientes:

- Valorar: Del 1 al 10 con el componente Bootstrap Star Rating (http://plugins.krajee.com/star-rating). Al rellenarse la valoración se enviará al servidor por medio de una llamada ajax.

  Será obligatorio en caso de realizar un comentario.

- Dificultad: Del 1 al 10 con el componente Bootstrap Slider (http://www.eyecon.ro/bootstrap-slider).

- Título: Campo de texto con el título del comentario. Será obligatorio en caso de realizar un comentario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength (http://mimo84.github.io/bootstrap-maxlength/)

- Comentario: El contenido del comentario del usuario. Será obligatorio en caso de realizar un comentario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Botón comentar: Botón por defecto de Bootstrap con el glyphicon "glyphicon-comment" precediendo al texto. Al hacer click sobre este botón se validará que el formulario es correcto y se enviarán el comentario y la valoración del usuario por medio de una llamada ajax.

- Compartir:

  Únicamente contiene el glyphicon "glyphicon-share".

  Al pasar por encima con el ratón se mostrará, con el componente tooltip de Bootstrap, el mensaje "Compartir".

  Al hacer clic se desplegará un footer del panel con todas las redes sociales en las que es posible compartir el tutorial con el componente jQuery Share In1 (http://plugins.in1.com/share).

- Ejecutar:

  Únicamente contiene el glyphicon "glyphicon-play".

  Al pasar por encima con el ratón se mostrará, con el componente tooltip de Bootstrap, el mensaje "Ejecutar".

  Al hacer clic se redireccionará a la pantalla de ejecución paso a paso de tutoriales.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la pantalla de presentación de tutoriales serán las siguientes:

- Búsqueda de tutoriales:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario. Alojada en la clase controlador de Spring ArticuloController, su especificación será la siguiente:

```
@RequestMapping(
           value="/**"
         , method = RequestMethod.GET)
   public String buscarArticulos(
           @ModelAttribute("filtro") Articulo pFiltro
         , Model model
         , Locale locale
         , Principal principal) {
   //TODO
}
```

  Esta función deberá llamar a la función fncBuscarArticulosNombrePasos del servicio ArticuloService para obtener los datos de los artículos publicados disponibles para la sección y nombre de artículo pasados por parámetro como parte del filtro de la pantalla. Una vez metida la lista de artículos en el modelo de Spring redireccionará a la vista de la pantalla de presentación de tutoriales.

- Valorar un tutorial:

  Esta función deberá ser llamada cuando se seleccionen las estrellas del componente de rating para el tutorial. Alojada en la clase controlador de Spring ArticuloController, su especificación será la siguiente:

```
@RequestMapping(
        value="/rate"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String valorarArticulo(
        @RequestParam("codArticulo") Integer pCodArticulo
      , @RequestParam("rate") Integer rate
      , Model model
      , Locale locale
      , Principal principal) {
   //TODO
}
```

  Esta función deberá llamar al método rate del servicio ArticuloService con el código del artículo y rate pasados por parámetros y el nombre del usuario logueado en el sistema para que su valoración se guarde en la base de datos.

- Comentar un tutorial:

  Esta función deberá ser llamada cuando haga click en el botón “Comentar”. Alojada en la clase controlador de Spring ArticuloController, su especificación

será la siguiente:

```
@RequestMapping(
        value="/comment"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String commentarArticulo(
        @RequestParam("codArticulo") Integer pCodArticulo
      , @RequestParam("dificultad") Integer dificultad
      , @RequestParam("resumen") String resumen
      , @RequestParam("comentario") String comentario
      , Model model
      , Locale locale
      , Principal principal) {
   //TODO
}
```

Esta función deberá llamar al método comment del servicio ArticuloService con el código del artículo, dificultad, título y comentario pasados por parámetros además del nombre del usuario logueado en el sistema para que su comentario se guarde en la base de datos.

En caso de que el comentario se guarde correctamente retornaremos a la función de ajax un mensaje indicando que el comentario se ha publicado correctamente.

Control de excepciones

Las excepciones que se den a la hora de navegar a otras pantallas se controlarán internamente por los handlers implementados a tal efecto en la arquitectura de la aplicación.

Por otro lado las excepciones que puedan darse a la hora de comentar y/o valorar un tutorial se mostrarán en un campo de feedback con el componente de alerts de tipo danger de Bootstrap (http://getbootstrap.com/components/#alerts). El mensaje de error será “Ups! Ha ocurrido un error inesperado, pruebe a intentarlo de nuevo más tarde.”

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que las secciones que aparecen en el combo son las que están publicadas en la base de datos (PUBLICADO = 1) y que aparecen aquellas que están en el idioma de la sesión (COD_IDIOMA = <idioma sesión>).
- Comprobar que la búsqueda funciona correctamente con los filtros de la pantalla.
- Comprobar que al pasar el ratón por encima de las opciones de los tutoriales se muestran los tooltips correspondientes.
- Validar que el botón de “Resumen” oculta cualquier footer ya desplegado y

despliega el footer con la lista de los títulos de todos los pasos del tutorial.

- Validar que el botón de “Lista de la compra” oculta cualquier footer ya desplegado y despliega el footer con la lista de componentes y sus cantidades del tutorial.
- Validar que si no se está logueado en la aplicación no aparece el botón “Comentar” y que sí se muestra en caso contrario.
- En caso de estar logueado en la aplicación, comprobar que el botón de “Comentar” oculta cualquier footer ya desplegado y despliega el footer con el formulario de valoración, comentario y los comentarios de otros usuarios.
- Valorar y comentar un tutorial comprobando que se valida correctamente el formulario y que se muestra el feedback de guardado.

  Volver a realizar una búsqueda del tutorial, pulsar en el botón de comentar y comprobar que el formulario está cumplimentado con los datos introducidos anteriormente.

  Desloguearse de la aplicación y loguearse con un usuario diferente, buscar el tutorial comentado y pulsar en “Comentar”. En la lista de comentarios de otros usuarios debería visualizarse el comentario y valoración introducidos anteriormente.
- Validar que el botón de “Compartir” oculta cualquier footer ya desplegado y despliega el footer con la lista redes sociales en las que compartir el tutorial. Verificar que se puede compartir correctamente el tutorial en todas ellas.
- Comprobar que el botón “Ejecutar” navega a la pantalla de ejecución del tutorial.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 8% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (8% sobre 1000 horas aprox.)	84 horas
Consumidas en diseño	5 horas
Disponibles para implementación	79 horas

Tabla 22: Presentación de tutoriales - Planificación

3.4.1.4. Ejecución paso a paso de un tutorial

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes al diagrama de Internet. En el análisis se especifica que un usuario puede:

- Ejecutar tutoriales.
- Valorar tutoriales.
- Comentar tutoriales.

Se trata de una pantalla sencilla que consta de una serie de diapositivas, tantas como pasos haya en el tutorial y una más en caso de que el usuario esté logueado en la aplicación para valorar y comentar el tutorial ejecutado al final de éste.

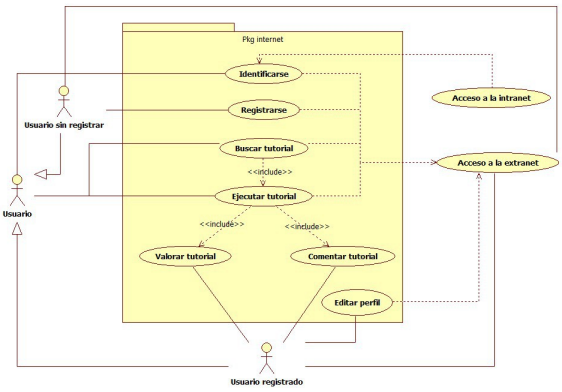


Ilustración 29: Ejecución paso a paso de un tutorial - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

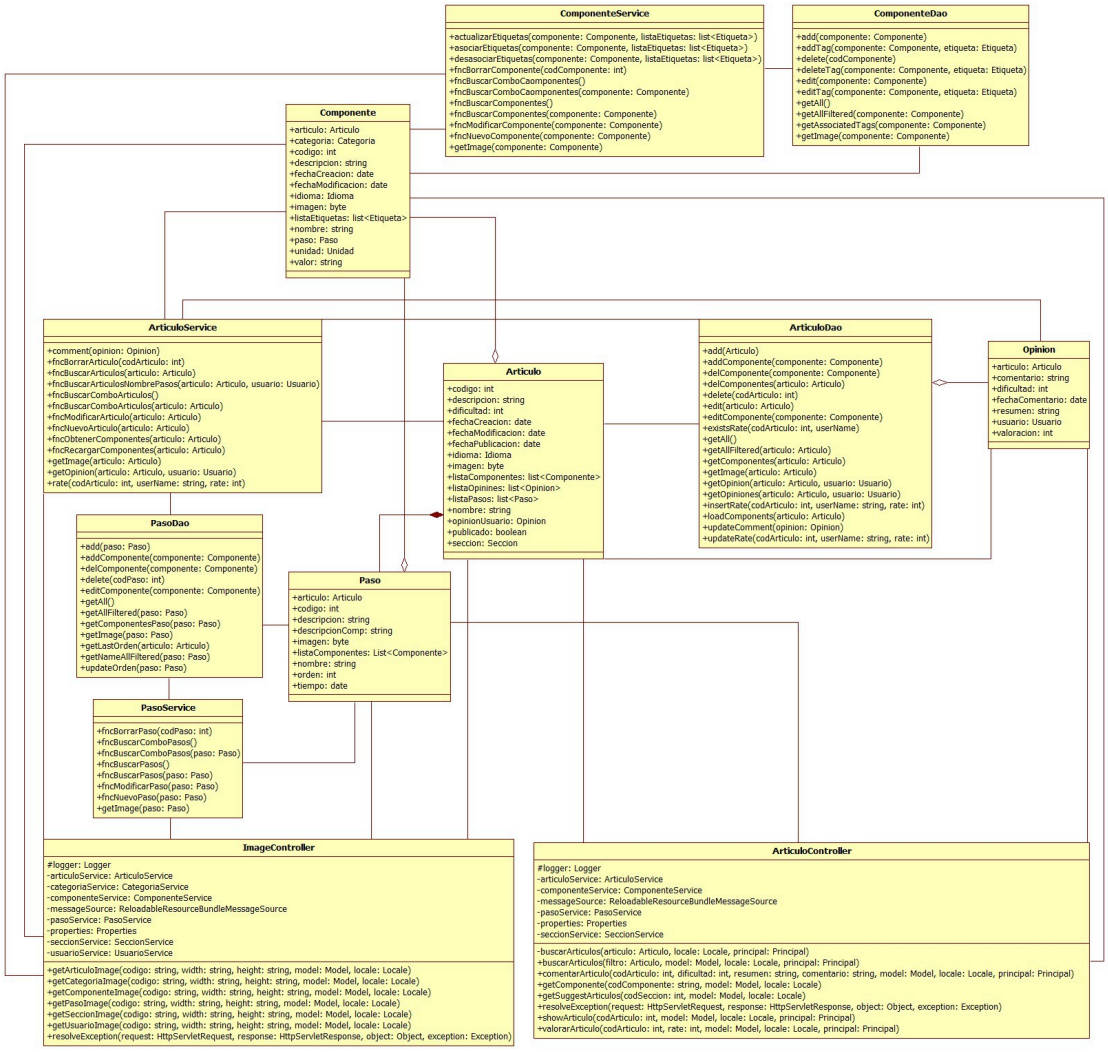


Ilustración 30: Ejecución paso a paso de un tutorial – Clases empleadas

Entidades empleadas:

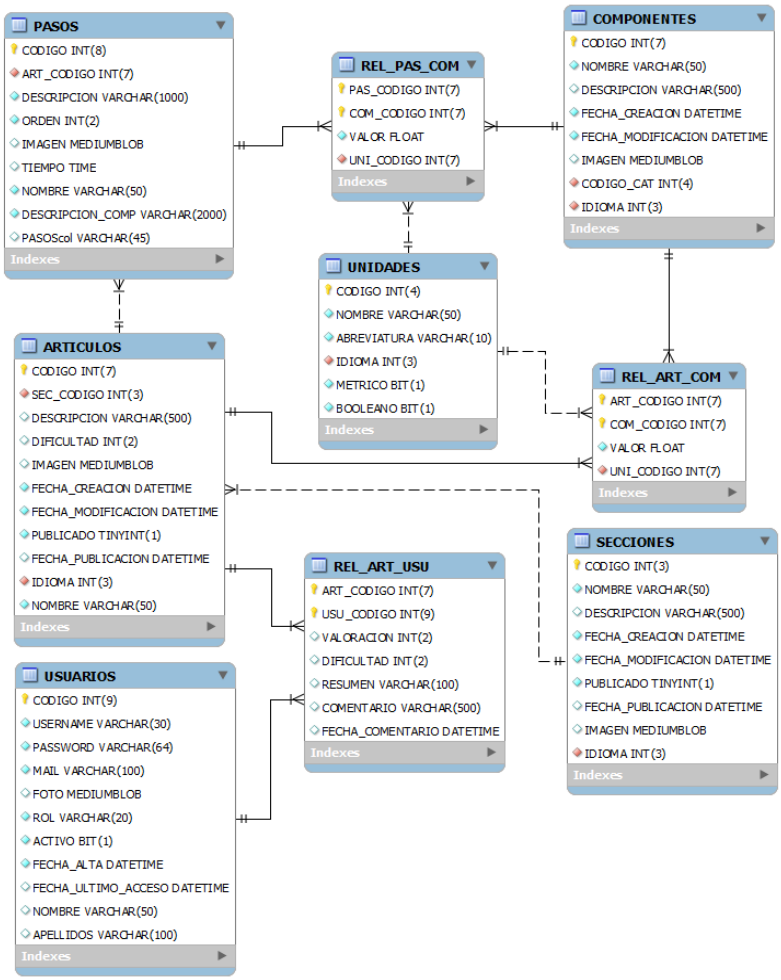


Ilustración 31: Ejecución de tutoriales - Entidades empleadas

Implementación de la seguridad

Cualquier tipo de usuario podrá acceder a la pantalla principal. Las particularidades de lo que verán los distintos perfiles se especificarán en el diseño de la pantalla.

Diseño de la pantalla

La pantalla de ejecución paso a paso varía el esquema de interfaz del resto de la aplicación puesto que la presentación de diapositivas ocupará toda la pantalla. Desaparecen la cabecera y el pie de página siendo reemplazados por los botones de navegación que serán exactamente los mismos tanto en la parte superior como inferior de la pantalla.

Estos botones de navegación serán los correspondientes a avanzar y retroceder entre diapositivas siguiendo el orden establecido de pasos dentro del tutorial.

También será posible desplazarse entre diapositivas deslizando con el dedo en dispositivos táctiles mediante el componente TouchSwype de jQuery.

Cada diapositiva presentada contendrá la siguiente información:

- Título del paso:

  Se mostrará de la forma "X/Y.- Título del paso". Donde X es el número del paso en el que nos encontramos e Y el número total de pasos que componen el tutorial.

- Imagen del paso:

  Muestra la imagen del paso como un enlace en el que al hacer clic se cargará la imagen en tamaño completo con el componente BlobkUI.

- Componentes del paso:

  Se visualiza una lista con los componentes y sus cantidades empleados en el paso.

- Descripción del paso:

  Muestra el texto del paso en sí mismo remarcando en él los componentes empleados con el componente label de Bootstrap de tipo default (http://getbootstrap.com/components/#labels). Estas etiquetas en el texto se comportarán como enlaces que al ser pulsados mostrarán con el componente popover de Bootstrap información referente al componente de la etiqueta.

  Para resoluciones pequeñas se sustituye el componente popover por el componente Bootstrap dialog para mostrar la información en una ventana modal.

En caso de que el usuario que está ejecutando el tutorial esté logueado en la aplicación, después del último paso se mostrará una diapositiva adicional para valorar y comentar el tutorial ejecutado. Ésta diapositiva contiene la siguiente información:

- Valorar: Del 1 al 10 con el componente Bootstrap Star Rating. Al rellenarse la valoración se enviará al servidor por medio de una llamada ajax.

  Será obligatorio en caso de realizar un comentario.

- Dificultad: Del 1 al 10 con el componente Bootstrap Slider.

- Título: Campo de texto con el título del comentario. Será obligatorio en caso de realizar un comentario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Comentario: El contenido del comentario del usuario. Será obligatorio en caso de realizar un comentario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Botón comentar: Botón por defecto de Bootstrap con el glyphicon “glyphicon-comment” precediendo al texto. Al hacer clic sobre este botón se validará que el formulario es correcto y se enviarán el comentario y valoración del usuario por medio de una llamada ajax.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la pantalla de ejecución de tutoriales serán las siguientes:

- Valorar un tutorial:

  Esta función deberá ser llamada cuando se seleccionen las estrellas del componente de rating para el tutorial. Alojada en la clase controlador de Spring ArticuloController, su especificación será la siguiente:

```
@RequestMapping(
        value="/rate"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String valorarArticulo(
        @RequestParam("codArticulo") Integer pCodArticulo
      , @RequestParam("rate") Integer rate
      , Model model
      , Locale locale
      , Principal principal) {
   //TODO
}
```

  Esta función deberá llamar al método rate del servicio ArticuloService con el código del artículo y rate pasados por parámetros además del nombre del usuario logueado en el sistema para que su valoración se guarde en la base de datos.

- Comentar un tutorial:

  Esta función deberá ser llamada cuando se haga click en el botón “Comentar”. Alojada en la clase controlador de Spring ArticuloController, su especificación será la siguiente:

```
@RequestMapping(
        value="/comment"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String commentarArticulo(
        @RequestParam("codArticulo") Integer pCodArticulo
      , @RequestParam("dificultad") Integer dificultad
      , @RequestParam("resumen") String resumen
      , @RequestParam("comentario") String comentario
      , Model model
      , Locale locale
      , Principal principal) {
   //TODO
}
```

Esta función deberá llamar al método comment del servicio ArticuloService con el código del artículo, dificultad, título y comentario pasados por parámetros además del nombre del usuario logueado en el sistema para que su comentario se guarde en la base de datos.

En caso de que el comentario se guarde correctamente retornaremos a la función de ajax un mensaje indicando que el comentario se ha publicado correctamente.

Control de excepciones

Las excepciones que puedan darse a la hora de comentar y/o valorar el tutorial se mostrarán en un campo de feedback con el componente de alerts de tipo danger de Bootstrap. El mensaje de error será: "Ups! Ha ocurrido un error inesperado, pruebe a intentarlo de nuevo más tarde."

Plan de pruebas

- Comprobar que los pasos que se muestran corresponden al tutorial ejecutado y que se encuentran en el orden correcto.
- Comprobar que al hacer clic en las fotos se muestran a tamaño completo.
- Comprobar que las listas de componentes y cantidades por paso son correctas.
- Validar que al pulsar sobre las etiquetas se muestra información de los componentes.
- Validar que si no se está logueado en la aplicación no aparece la diapositiva adicional para valorar y comentar el tutorial.
- Valorar y comentar un tutorial comprobando que valida correctamente el formulario y que se muestra el feedback de guardado.
- Volver a realizar una búsqueda del tutorial, pulsar en el botón de comentar y comprobar que el formulario está completo con los datos introducidos anteriormente. Ejecutar el tutorial, navegar a la pantalla de valoración y comentario y comprobar que está igualmente cumplimentado.
- Desloguearse de la aplicación y loguearse con un usuario diferente, buscar el tutorial comentado y pulsar en "Comentar". En la lista de comentarios de otros usuarios debería visualizarse el comentario y valoración introducidos anteriormente.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 5% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (5% sobre 1000 horas aprox.)	50 horas
Consumidas en diseño	3 horas
Disponibles para implementación	47 horas

Tabla 23: Ejecución paso a paso de un tutorial - Planificación

3.4.1.5. Administración

Caso o casos de uso a implementar

No implementa, es una página de transición.

Objetos del modelo y entidades de base de datos

Clases empleadas:

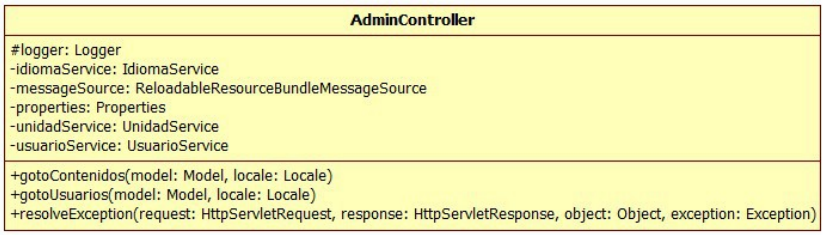


Ilustración 32: Administración - Diagrama de clases

Entidades empleadas:

No se requiere el uso de entidades de base de datos.

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página. Las particularidades de lo que verán los distintos perfiles se especificarán en el diseño de la pantalla.

Diseño de la pantalla

En esta pantalla se mostrarán dos grandes imágenes a modo de botones:

- Gestión de usuarios.

  Se mostrará únicamente si el usuario registrado tiene perfil de administrador. Al hacer clic navegará a la pantalla de gestión de usuarios.

- Gestión de contenidos.

  Al hacer clic se navegará a la pantalla de gestión de contenidos.

Diseño funcional

No hay diseño funcional, solamente presentación.

Control de excepciones

Las excepciones que se den a la hora de navegar a otras pantallas se controlarán internamente por los handlers implementados a tal efecto en la arquitectura de la aplicación.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario con perfil de mánager únicamente se muestra el enlace a la gestión de contenidos.

  Verificar que el enlace navega correctamente a la pantalla de gestión de contenidos.

- Comprobar que al entrar por URL estando logueado con un usuario con perfil de administrador se muestran los dos enlaces.

  Verificar que ambos enlaces navegan correctamente a sus correspondientes pantallas.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 1% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (1% sobre 1000 horas aprox.)	8 horas
Consumidas en diseño	2 horas
Disponibles para implementación	6 horas

Tabla 24: Administración - Planificación

3.4.1.6. Gestión de usuarios

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de usuarios. Donde en el análisis se especifica que un usuario puede:

- Crear usuario.
- Modificar usuario.
- Eliminar usuario.

Aunque un administrador pueda realizar modificaciones sobre los usuarios, las contraseñas de éstos en ningún momento le serán visibles. Podrá restaurarlas por unas nuevas, pero nunca visualizarlas.

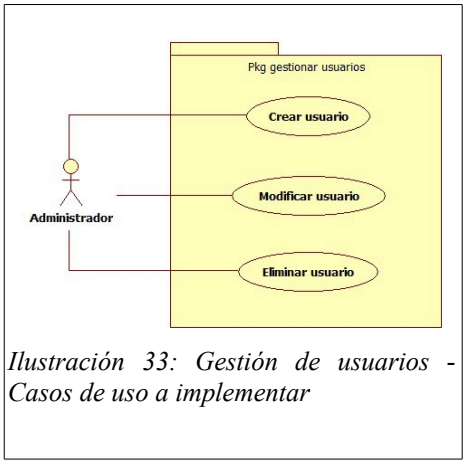


Ilustración 33: Gestión de usuarios - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

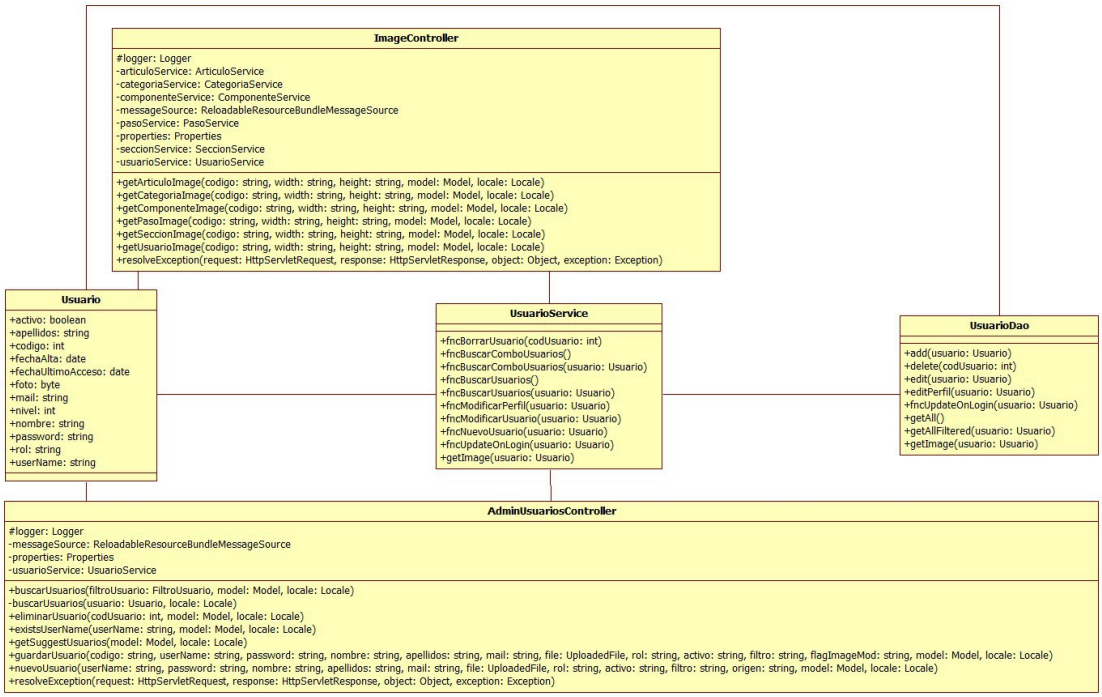


Ilustración 34: Gestión de usuarios – Clases empleadas

Entidades empleadas:

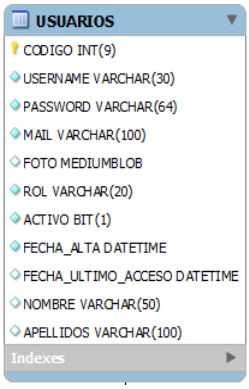


Ilustración 35: Diseño de la cabecera - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador tendrán acceso a esta página.

Diseño de la pantalla

La pantalla mostrará dos artículos en la sección. Uno de ellos contendrá el formulario de búsqueda de usuarios y el otro la tabla de resultados encontrados.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Nombre del usuario: Campo de texto simple.
- Botón de buscar: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-search". Al hacer clic sobre este botón se enviará el formulario por método POST para buscar usuarios con el nombre del filtro.
- Botón nuevo: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-plus". Al hacer clic sobre este botón se mostrará un formulario en una ventana modal creada con el componente Bootstrap Dialog cuyo título será "Nuevo usuario" y contendrá un formulario con los siguientes campos:
  - Usuario:

    Campo de texto obligatorio.
  - Contraseña:

    Campo de texto de tipo password obligatorio.
  - Confirmación de contraseña:

    Campo de texto de tipo password obligatorio que validará si es exacto al campo anterior de contraseña.
  - Nombre:

    Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Apellidos:

    Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - E-mail:

    Campo de texto obligatorio que validará si es una dirección de e-mail válida. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Imagen:

  Input de tipo file. No es obligatorio.

- Rol:

  Combo con los roles en la aplicación: en blanco (usuario normal), administrador y mánager.

- Activo:

  Checkbox que indica si el usuario puede loguearse en la aplicación o en su defecto está bloqueado. Por defecto se mostrará seleccionado.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente y en caso satisfactorio lanzará la acción de registrar un nuevo usuario.

- Botón de cancelar:

  Será un botón de tamaño normal por defecto de Bootstrap. Cancela el alta de usuario cerrando el diálogo con el formulario.

En el segundo artículo se muestra la tabla de resultados formada con el componente FooTable (http://themergency.com/footable/) con los siguientes campos visibles en sus diferentes configuraciones:

- Escritorio: Imagen, usuario, rol, nombre, apellidos, e-mail, fecha de alta y último acceso.
- Tableta: Imagen, usuario, rol y fecha de alta.
- Móvil: Usuario y rol.
- Siempre oculto: Botones de editar y borrar.

Se ordenará por defecto por la columna de usuario de forma ascendente, siendo posible la ordenación por cualquier columna salvo la de imagen.

Las imágenes de usuario serán enlaces que mostrarán la foto del usuario en tamaño completo mediante el componente BlockUI.

Al pulsar en el botón “Editar” se desplegará de la fila un formulario con los siguientes datos:

- Usuario:

  Campo de texto obligatorio. Estará cargado con el username del usuario.

- Contraseña:

  Se mostrará un botón por defecto de Bootstrap con el texto “Cambiar contraseña”. En caso de ser pulsado el botón se ocultará y en su lugar aparecerán

dos campos de texto obligatorios de tipo password: “Contraseña” y “Confirmación”. El campo de confirmación validará si es exacto al campo anterior de contraseña.

- Nombre:

  Campo de texto obligatorio. Estará cargado con el nombre del usuario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Apellidos:

  Campo de texto obligatorio. Estará cargado con los apellidos del usuario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- E-mail:

  Campo de texto obligatorio que validará si es una dirección de e-mail válida. Estará cargado con el e-mail del usuario. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Imagen:

  Input de tipo file si el usuario no tenía imagen guardada o un botón por defecto de Bootstrap con el texto “Cambiar imagen”. Al pulsar en este botón se ocultará y aparecerá un input de tipo file para añadir la nueva imagen. No es obligatorio. Si modificamos y no añadimos una imagen nueva borraremos la que ya estaba guardada en la base de datos.

- Rol:

  Combo con los roles en la aplicación: en blanco (usuario normal), administrador y mánager. Estará seleccionado el perfil del usuario.

- Activo:

  Checkbox que indica si el usuario puede loguearse en la aplicación o en su defecto está bloqueado. Estará seleccionado si el usuario no está bloqueado.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente y en caso satisfactorio lanzará la acción de modificar usuario.

- Botón de cancelar:

  Será un botón de tamaño normal tipo danger de Bootstrap. Cancela las modificaciones realizadas en el usuario y oculta el formulario de edición.

No se podrán tener dos formularios de edición abiertos. Si se va a abrir un segundo formulario de edición se cerrará el que esté abierto en ese momento de forma automática. Si se da el caso de que el formulario a cerrar automáticamente tenía cambios sin guardar se preguntará al administrador si desea descartarlos antes de

continuar o por lo contrario prefiere cancelar la edición del segundo usuario.

Al pulsar el botón “Borrar” se llamará por ajax a la función de borrado de usuarios. Si el borrado se ha realizado correctamente se elimina la fila de la tabla con el usuario.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la gestión de usuarios serán las siguientes:

- Búsqueda de usuarios:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminUsuariosController, su especificación será la siguiente:

```
@RequestMapping(value = "buscarUsuarios", method = RequestMethod.POST)
   public String buscarUsuarios(
           @ModelAttribute("usuarioFiltro") FiltroUsuario usuarioFiltro
         , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar al método fncBuscarUsuarios del servicio UsuarioService para obtener los datos de los usuarios que coincidan con el filtro de la pantalla. Una vez metida la lista de usuarios en el modelo de Spring redireccionará a la pantalla de gestión de usuarios y mostrará los resultados.

- Alta de usuario:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar” del formulario de alta de usuario. Alojada en la clase controlador de Spring AdminUsuariosController, su especificación será la siguiente:

```
@RequestMapping(value = "nuevoUsuario", method = RequestMethod.POST)
   public String nuevoUsuario(
           @ModelAttribute("userName") String userName
         , @ModelAttribute("password") String password
         , @ModelAttribute("nombre") String nombre
         , @ModelAttribute("apellidos") String apellidos
         , @ModelAttribute("mail") String mail
         , @ModelAttribute("FORM") UploadedFile file
         , @ModelAttribute("rol") String rol
         , @ModelAttribute("activo") String activo
         , @ModelAttribute("filtro") String filtro
         , @ModelAttribute("origen") String origen
         , @ModelAttribute("searched") String searched
         , Model model, Locale locale
         , HttpServletResponse response) {
   //TODO
}
```

  Esta función deberá llamar al método fncNuevoUsuario del servicio UsuarioService para insertar el usuario con los parámetros recibidos por

pantalla. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que el usuario se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de usuario:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario edición del usuario. Alojada en la clase controlador de Spring AdminUsuariosController, su especificación será la siguiente:

```
@RequestMapping(value = "guardarUsuario", method = RequestMethod.POST)
   public String guardarUsuario(
        @ModelAttribute("codigo") String codigo
      , @ModelAttribute("userName") String userName
      , @ModelAttribute("password") String password
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("apellidos") String apellidos
      , @ModelAttribute("mail") String mail
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("rol") String rol
      , @ModelAttribute("activo") String activo
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("flagImageMod") String flagImageMod
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar al método fncModificarUsuario del servicio UsuarioService para actualizar el usuario con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que el usuario se ha guardado correctamente usando el componente alerts de tipo success de Bootstrap.

- Borrado de usuario:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Borrar" de la fila de usuario seleccionada. Alojada en la clase controlador de Spring AdminUsuariosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarUsuario"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarUsuario(
      @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar al método fncBorrarUsuario del servicio UsuarioService con el código de usuario recibidos por pantalla para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que el usuario se ha eliminado correctamente usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de usuarios se mostrarán en la parte superior de la sección (encima del artículo con los criterios de búsqueda) usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario con perfil mánager redirige a la página de error con un mensaje de falta de permisos.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Redimensionar la pantalla y comprobar que el componente de tabla cambia correctamente.
- Comprobar que al dar de alta un usuario: el formulario se valida correctamente, los datos se guardan bien en base de datos y aparece el mensaje de confirmación en pantalla.
- Comprobar que al pulsar el botón “Editar” se despliega el formulario de edición cumplimentado correctamente.
- Comprobar que al pulsar el botón “Cambiar contraseña”, éste desaparece y se muestran los campos de contraseña.
- Comprobar que al pulsar el botón “Cambiar imagen”, éste desaparece y se muestra el campo para insertar imagen.
- Modificar los datos de un usuario, pulsar “Guardar” y comprobar que los datos se actualizan correctamente en la base de datos. Validar que se muestra el feedback en pantalla.
- Pulsar “Borrar” y comprobar que el usuario se ha eliminado de base de datos. Validar que ha desaparecido de la tabla y que se muestra el feedback en la pantalla.
- Pulsar en “Editar” para modificar los datos de un usuario y después pulsar en el “Editar” de otro usuario. Comprobar que sólo se muestra el formulario de edición del segundo usuario.
- Pulsar en “Editar” para modificar los datos de un usuario, realizar

modificaciones y después pulsar en “Editar” de otro usuario. Comprobar que se muestra un diálogo de confirmación para deshacer los cambios del primer usuario antes de continuar.

- Modificar los datos de un usuario, pulsar “Cancelar” y comprobar que se cancelan las modificaciones correctamente.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 7% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (7% sobre 1000 horas aprox.)	67 horas
Consumidas en diseño	7 horas
Disponibles para implementación	60 horas

Tabla 25: Gestión de usuarios - Planificación

3.4.1.7. Gestión de contenidos

Caso o casos de uso a implementar

Los casos de uso a implementar serán los referentes a la gestión de contenidos. Concretamente en esta pantalla podrán ejecutarse los siguientes:

- Nueva sección.
- Nueva categoría.
- Nuevo componente.
- Nueva etiqueta.

Por lo demás, esta pantalla es un índice de las diferentes pantallas de gestión de contenidos.

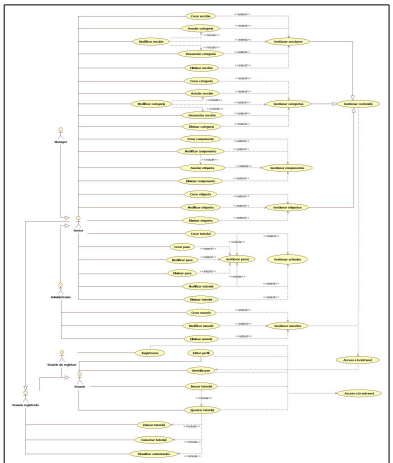


Ilustración 36: Gestión de contenidos – Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

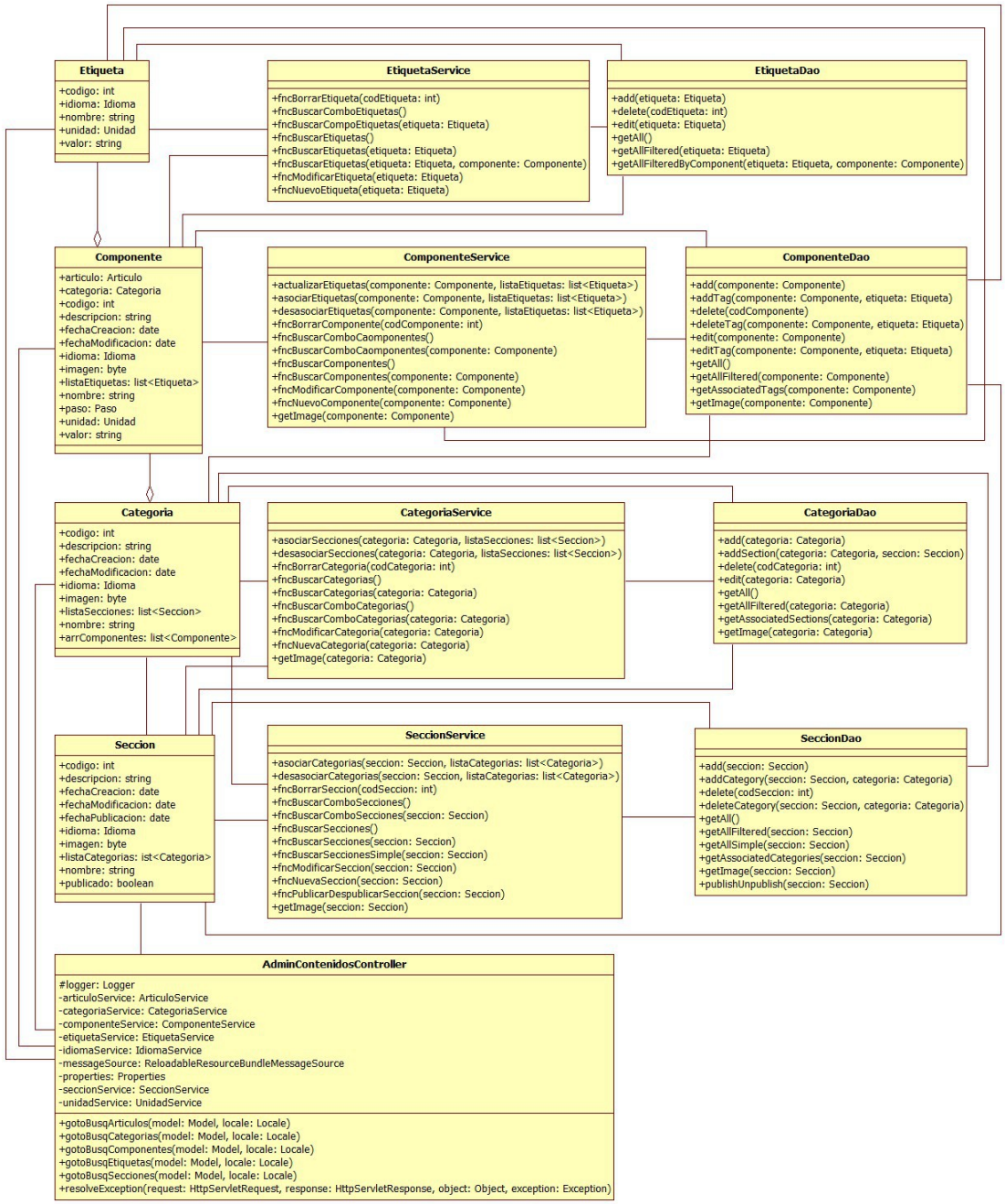


Ilustración 37: Gestión de contenidos – Clases empleadas

Entidades empleadas:

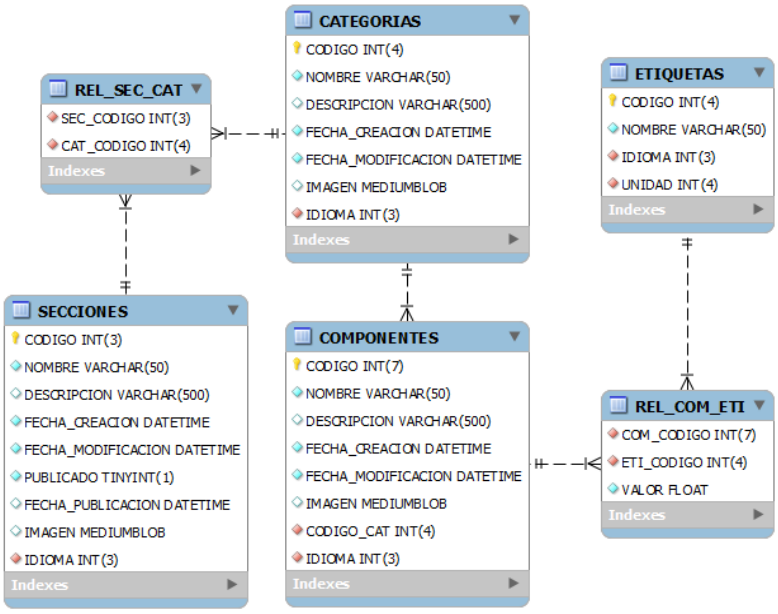


Ilustración 38: Gestión de contenidos - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página.

Diseño de la pantalla

Se presentarán los siguientes botones por defecto de Bootstrap en pantalla:

- Secciones:

  Al hacer clic navegará a la pantalla de gestión de secciones.

- Nueva sección:

  Se abrirá una ventana modal con el componente Bootstrap Dialog mostrando el formulario de alta de secciones. El formulario contendrá los siguientes campos:

  - Idioma: Combo obligatorio con los idiomas disponibles.
  - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Descripción: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Imagen: Campo de tipo fichero.

- Publicado: Campo de tipo checkbox.
- Guardar: Botón de tipo principal de Bootstrap que al pulsar valida el formulario y en caso de ser correcto lo envía.
- Cancelar: Botón por defecto de Bootstrap que al pulsar cancela el alta cerrando la ventana modal.

- Categorías:

  Navega a la pantalla de gestión de categorías.

- Nueva categoría:

  Se abrirá una ventana modal con el componente Bootstrap Dialog mostrando el formulario de alta de secciones. El formulario contendrá los siguientes campos:

  - Idioma: Combo obligatorio con los idiomas disponibles.
  - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Descripción: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Imagen: Campo de tipo fichero.
  - Guardar: Botón de tipo principal de Bootstrap que al pulsar valida el formulario y en caso de ser correcto lo envía.
  - Cancelar: Botón por defecto de Bootstrap que al pulsarlo cancela el alta y cierra la ventana modal.

- Componentes:

  Navega a la pantalla de gestión de componentes.

- Nuevo componente:

  Se abrirá una ventana modal con el componente Bootstrap Dialog mostrando el formulario de alta de secciones. El formulario contendrá los siguientes campos:

  - Idioma: Combo obligatorio con los idiomas disponibles.
  - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Descripción: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Imagen: Campo de tipo fichero.
  - Categoría: Campo de texto simple deshabilitado con la categoría asociada. Este campo es obligatorio.

- Asociar categoría: Botón por defecto de Bootstrap que muestra una segunda ventana modal creada con el componente Bootstrap Dialog que enseña una búsqueda de categorías. En esta ventana se mostrará:
  - Un campo de texto simple para filtrar las categorías por nombre.
  - Una tabla creada con el componente FooTable que contendrá todas las categorías dadas de alta.

    Las filas de la tabla mostrarán un radio de selección y el nombre de la categoría. Como campo oculto tendremos la descripción de la misma.
  - Un botón de Bootstrap de tipo principal con el texto "Asociar". Al pulsarse se cerrará la ventana y en caso de haberse seleccionado alguna categoría ésta aparecerá en el campo de texto de categoría.
  - Un botón por defecto de Bootstrap con el texto "Cancelar". Al pulsarse se cerrará la ventana.
- Guardar: Botón de tipo principal de Bootstrap que al pulsar valida el formulario y en caso de ser correcto lo envía.
- Cancelar: Botón por defecto de Bootstrap que al pulsarlo cancela el alta cerrando la ventana modal.

- Etiquetas:

  Navega a la pantalla de gestión de etiquetas.
- Nueva etiqueta:

  Se abrirá una ventana modal con el componente Bootstrap Dialog mostrando el formulario de alta de etiquetas. El formulario contendrá los siguientes campos:
  - Idioma: Combo obligatorio con los idiomas disponibles.
  - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
  - Unidad: Combo obligatorio con las unidades disponibles. Su contenido dependerá de lo seleccionado en el combo de idioma.
- Tutoriales:

  Navega a la pantalla de gestión de tutoriales.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la pantalla serán las siguientes:

- Alta de secciones:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar"

del formulario de alta de sección. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevaSeccion"
      , method = RequestMethod.POST)
   public String nuevaSeccion(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("publicado") String publicado
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevaSeccion del servicio SeccionService para insertar la sección con los parámetros recibidos por pantalla. En caso de que en la base de datos la operación se realice correctamente se mostrará en pantalla que la sección se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Alta de categorías:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario de alta de categoría. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevaCategoria"
      , method = RequestMethod.POST)
   public String nuevaCategoria(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevaCategoria del servicio CategoriaService para insertar la categoría con los parámetros recibidos por pantalla. En caso de que en la base de datos la operación se realice correctamente se mostrará en pantalla que la categoría se ha guardado correctamente usando el componente alerts de tipo success de Bootstrap.

- Alta de componentes:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario de alta de componente. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevoComponente"
      , method = RequestMethod.POST)
   public String nuevoComponente(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , @ModelAttribute("searched") String searched
      , @ModelAttribute("codCategoria") Integer codCategoria
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevoComponente del servicio ComponenteService para insertar el componente con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que el componente se ha guardado correctamente usando el componente alerts de tipo success de Bootstrap.

- Alta de etiquetas:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario de alta de etiqueta. Alojada en la clase controlador de Spring AdminEtiquetasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevoEtiqueta"
      , method = RequestMethod.POST)
   public String nuevoEtiqueta(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("unidad") String unidad
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , @ModelAttribute("searched") String searched
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevoEtiqueta del servicio EtiquetaService para insertar la etiqueta con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que la etiqueta se ha guardado correctamente usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que se den a la hora de navegar a otras pantallas se controlarán internamente por los handlers implementados a tal efecto en la arquitectura de la aplicación.

Las excepciones que ocurran durante las diferentes altas que pueden realizarse desde la pantalla se mostrarán en la parte superior de la sección usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que el botón "Secciones" navega correctamente a la pantalla de gestión de secciones.
- Comprobar que al dar de alta una sección: el formulario se valida correctamente, los datos se guardan bien en base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que el botón "Categorías" navega correctamente a la pantalla de gestión de categorías.
- Comprobar que al dar de alta una categoría: el formulario se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que el botón "Componentes" navega correctamente a la pantalla de gestión de componentes.
- Comprobar que al dar de alta un componente: el formulario se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que el botón "Etiquetas" navega correctamente a la pantalla de gestión de etiquetas.
- Comprobar que al dar de alta una etiqueta: el formulario se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que el botón "Tutoriales" navega correctamente a la pantalla de gestión de tutoriales.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 3% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (3% sobre 1000 horas aprox.)	25 horas
Consumidas en diseño	5 horas
Disponibles para implementación	20 horas

Tabla 26: Gestión de contenidos – Planificación

3.4.1.8. Gestión de secciones

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de secciones, donde en el análisis se especifica que un usuario puede:

- Crear sección
- Asociar categoría
- Desasociar categoría
- Modificar sección
- Eliminar sección

Las secciones son el nivel principal de catalogación de los tutoriales (recetas, bricolaje, etc.) y no se prevé que vaya a haber un gran número de ellas dadas de alta en el sistema a largo plazo.

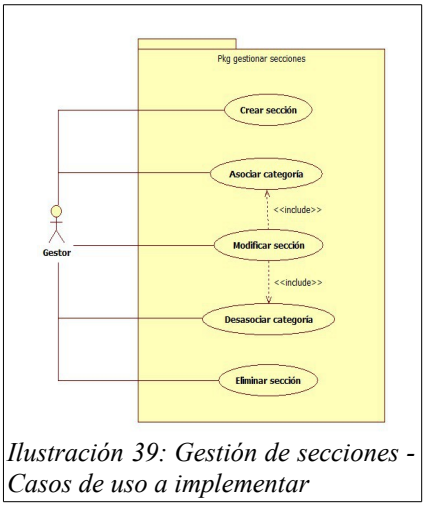


Ilustración 39: Gestión de secciones - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

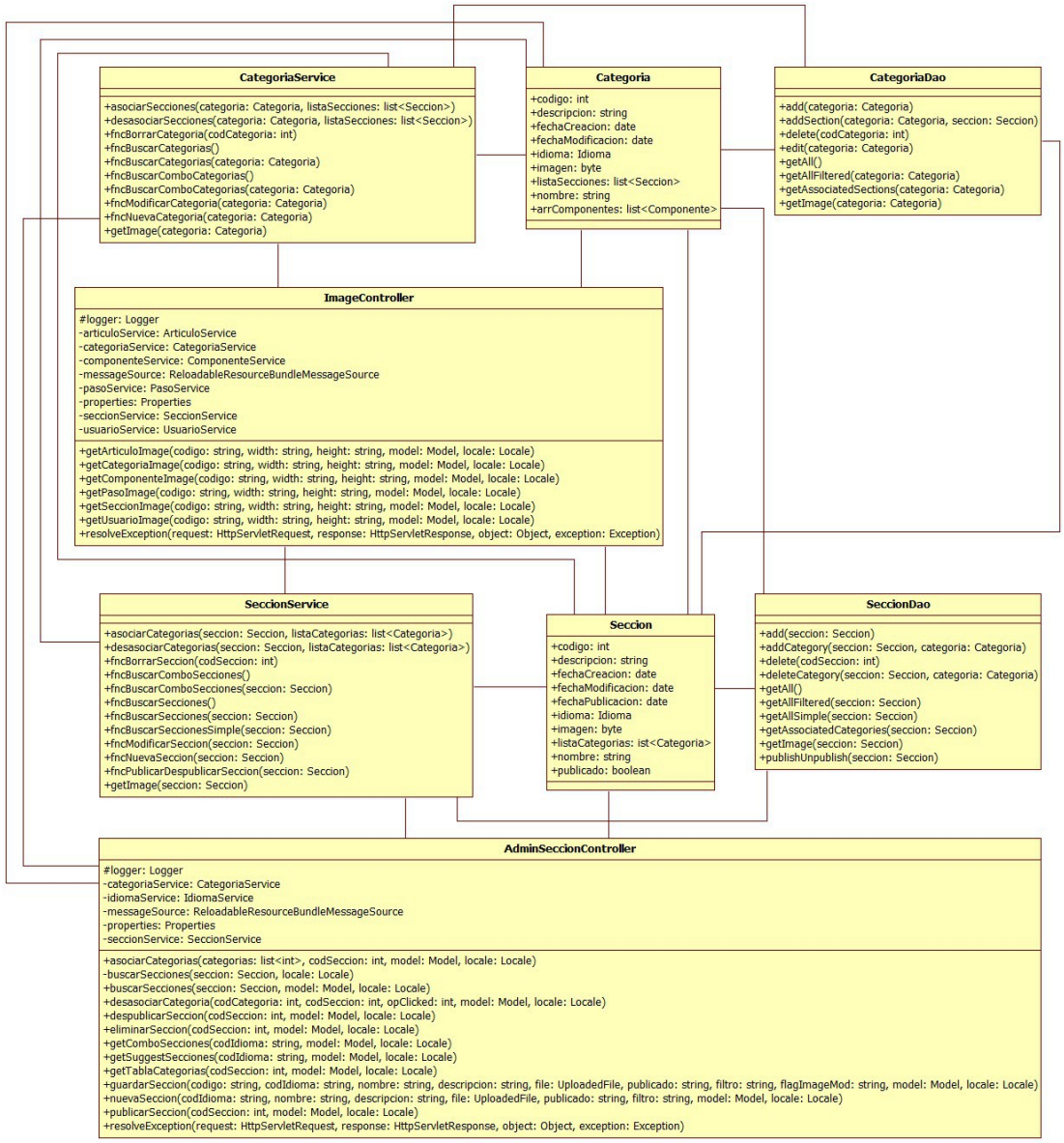


Ilustración 40: Gestión de secciones – Clases empleadas

Entidades empleadas:

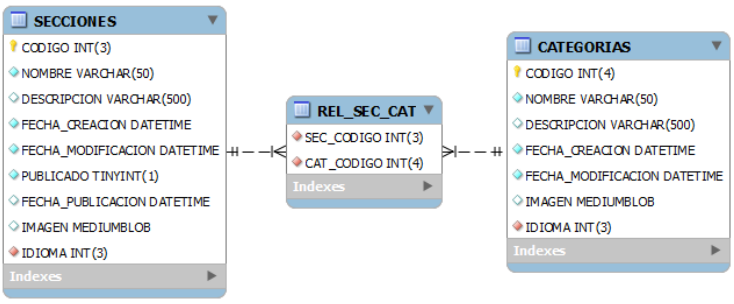


Ilustración 41: Gestión de secciones - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página.

Diseño de la pantalla

La pantalla mostrará varios artículos en la sección. El primero de ellos contendrá el formulario de búsqueda de secciones. Cada uno de los siguientes se corresponderá con una de las secciones encontradas al ejecutar la búsqueda.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Idioma: Combo cargado con los idiomas disponibles.
- Sección: Campo de texto simple para filtrar las secciones por nombre. Se requiere que este campo sugiera secciones que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI.
- Botón de buscar: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-search". Al hacer clic sobre este botón se enviará el formulario por método POST para buscar secciones que coincidan con el filtro.
- Botón nuevo: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-plus". Al hacer clic sobre este botón se mostrará un formulario en una ventana modal, creada con el componente Bootstrap Dialog, cuyo título será "Nueva sección" y contendrá un formulario con los siguientes campos:
  - Idioma: Combo obligatorio con los idiomas disponibles.
  - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Descripción: Textarea obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
- Imagen: Campo de tipo fichero.
- Publicado: Campo de tipo checkbox que chequeado tomará el valor “S”.

Cada artículo de resultado de la búsqueda mostrará la siguiente información:

- Nombre: El nombre de la sección.
- Descripción: La descripción de la sección.
- Imagen: Imagen de la sección. Será un enlace que al pulsarse, cargará la imagen a tamaño completo en la pantalla con el componente BlockUI.
- Menú de opciones: En la parte superior derecha del artículo se mostrará un menú de opciones a realizar. Se detallan a continuación:
  - Editar:

    Botón por defecto de Bootstrap sin texto y con el glyphicon “glyphicon-pencil”.

    Al pulsar este botón se desplegará un formulario de edición de la sección debajo de los datos mostrados en la misma. Este formulario contendrá los siguientes campos:

    - Fecha creación: Campo de texto simple deshabilitado y precargado con la fecha de creación de la sección.
    - Fecha modificación: Campo de texto simple deshabilitado y precargado con la fecha de la última modificación de la sección.
    - Idioma: Combo obligatorio con todos los idiomas disponibles y precargado con el idioma de la sección.
    - Nombre: Campo obligatorio de texto simple y precargado con el nombre de la sección. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Descripción: Textarea obligatorio y precargado con la descripción de la sección. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Imagen: Campo de tipo fichero.
    - Publicado: Campo de tipo checkbox que chequeado tomará el valor “S”. Estará chequeado si la sección se encuentra publicada.
    - Fecha publicación: Campo de texto simple deshabilitado y precargado con la fecha de la última publicación de la sección, en caso de estar publicada.

- Categorías: Combo precargado con las categorías asociadas a la sección.
- Asociar categoría: Botón por defecto de Bootstrap que tiene un span a la izquierda del texto con el glyphicon “glyphicon-plus-sign”. Al hacer clic sobre este botón se mostrará un formulario en una ventana modal, creada con el componente Bootstrap Dialog, cuyo título será “Asociar categoría” que muestra una búsqueda de categorías. En esta ventana se mostrará:
  - Un campo de texto simple para filtrar las categorías por nombre.
  - Una tabla creada con el componente FooTable que contendrá todas las categorías dadas de alta en el idioma de la sección.

    Las filas de la tabla mostrarán un checkbox de selección y el nombre de la categoría. Como campo oculto tendremos la descripción de la misma. El campo de selección aparecerá chequeado en caso de que la categoría se encuentre asociada a la sección que se está editando.
  - Un botón de Bootstrap de tipo principal con el texto “Asociar”. Al pulsarse, se cerrará la ventana y lanzará una operación de ajax con las categorías seleccionadas para que se guarden en la base de datos. Esta operación de ajax retornará las categorías con sus nombres para actualizar el combo “Categorías” de la sección.

    En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.
  - Un botón por defecto de Bootstrap con el texto “Cancelar”. Al pulsarse se cerrará la ventana.
- Desasociar categoría: Desasocia de la sección editada la categoría seleccionada en el combo “Categorías”. Se realizará por medio de una operación de ajax.

  En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.

Al pulsarse el botón de edición, el artículo de la sección pasará a estar en modo edición y el botón “Editar” se ocultará en favor de otros dos botones:

- Guardar: Botón de tipo principal de Bootstrap sin texto y con el glyphicon “glyphicon-floppy-disk”. Al pulsar este botón se comprueba que el formulario de edición sea válido. En ese caso se envía al servidor.
- Cancelar: Botón por defecto de Bootstrap sin texto y con el glyphicon “glyphicon-floppy-remove”. Al pulsar este botón se colapsa el formulario de edición y se restaura deshaciéndose los cambios.

No se podrán tener dos formularios de edición abiertos. Si se va a abrir un segundo formulario de edición se cerrará el que esté abierto en ese momento de forma automática. En caso de que el formulario previamente abierto tuviese cambios sin guardar, se preguntará al usuario si desea descartarlos antes de continuar o por lo contrario prefiere cancelar la edición del segundo formulario.

- Borrar:

  Botón de tipo danger de Bootstrap sin texto y con el glyphicon "glyphicon-trash".

  Por medio de una operación de ajax se realizará el borrado en la base de datos de la sección y sus relaciones con las diferentes categorías.

  En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.

- Publicar/Despublicar:

  Este botón en realidad son dos, "Publicar" y "Despublicar", únicamente se mostrará uno de los dos en función de si la sección está publicada.

  En caso de estar publicada la sección, se mostrará el botón "Despublicar". Botón por defecto de Bootstrap sin texto y con el glyphicon "glyphicon-eye-close". Este botón lanza una operación de ajax encargada de modificar la sección marcándola como no publicada en la base de datos.

  En caso de estar despublicada la sección, se mostrará el botón "Publicar". Botón por defecto de Bootstrap sin texto y con el glyphicon "glyphicon-eye-open". Este botón lanza una operación de ajax encargada de modificar la sección marcándola como publicada en la base de datos.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la gestión de secciones serán las siguientes:

- Búsqueda de secciones:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "buscarSecciones"
      , method = RequestMethod.POST)
   public String buscarSecciones(
        @ModelAttribute("filtro") Seccion filtro
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncBuscarSecciones del servicio SeccionService para obtener los datos de las secciones para el nombre de la sección pasado por parámetro como parte del filtro de la pantalla. Una vez metida la lista de secciones en el modelo de Spring, redireccionará a la pantalla de gestión de secciones y mostrará los resultados.

En el siguiente diagrama de secuencia se muestra una operación con envío de fomulario de filtro con Spring a través de esta operación de búsqueda de secciones:

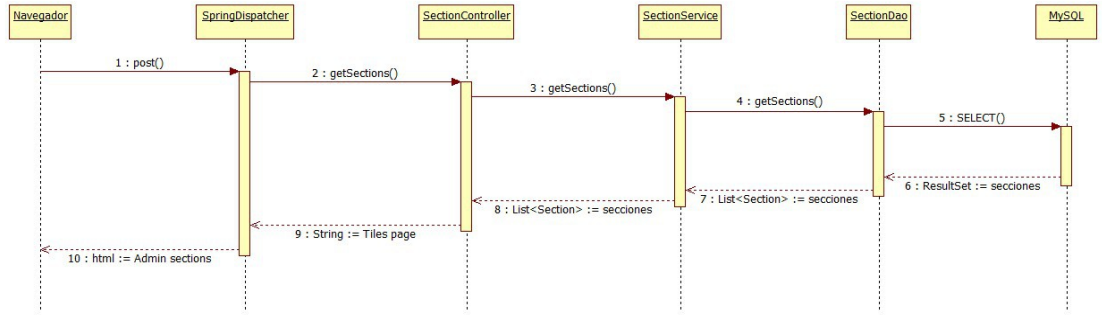


Ilustración 42: Diagrama de secuencia - Submit de formulario en Spring

- Alta de sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario de alta de sección. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevaSeccion"
      , method = RequestMethod.POST)
   public String nuevaSeccion(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("publicado") String publicado
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevaSeccion del servicio SeccionService para insertar la sección con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la sección se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario edición de la sección. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarSeccion"
      , method = RequestMethod.POST)
   public String guardarSeccion(
        @ModelAttribute("codigo") String codigo
      , @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("publicado") String publicado
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("flagImageMod") String flagImageMod
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncModificarSeccion del servicio SeccionService para actualizar la sección con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la sección se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Publicar sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Publicar" de la sección seleccionada. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "publicarSeccion"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String publicarSeccion(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncPublicarDespublicarSeccion del servicio SeccionService con el código de sección recibido por la pantalla para actualizarla en la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la sección se ha

publicado usando el componente alerts de tipo success de Bootstrap.

En el siguiente diagrama de secuencia se muestra una operación ajax con Spring a través de esta operación de publicado de secciones:

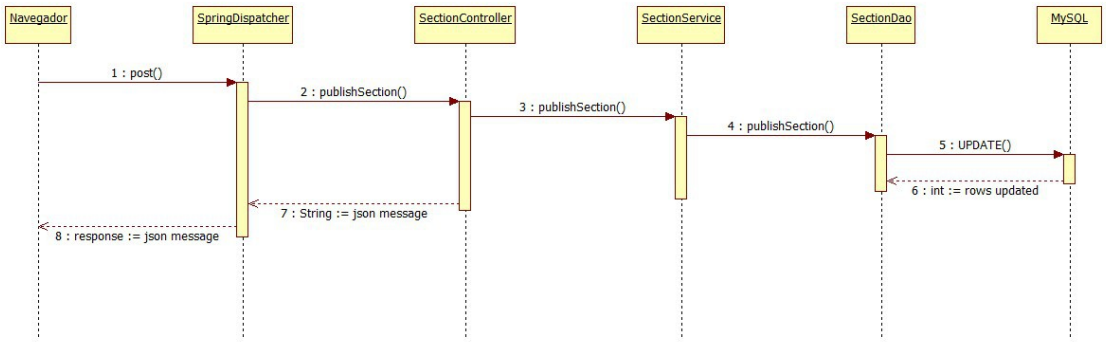


Ilustración 43: Diagrama de secuencia - Operación ajax en Spring

- Despublicar sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Despublicar" de la sección seleccionada. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "despublicarSeccion"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String despublicarSeccion(
       @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncPublicarDespublicarSeccion del servicio SeccionService, con el código de sección recibido por la pantalla, para actualizarla en la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la sección se ha publicado usando el componente alerts de tipo success de Bootstrap.

- Borrado de sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Borrar" de la sección seleccionada. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarSeccion"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarSeccion(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncBorrarSeccion del servicio SeccionService, con el código de sección recibido por pantalla, para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en pantalla que la sección se ha eliminado usando el componente alerts de tipo success de Bootstrap.

- Asociar categorías:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Asociar categoría" del formulario de edición de la sección seleccionada. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "asociarCategorias"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String asociarCategorias(
        @ModelAttribute("categorias") List<Integer> pCategorias
      , @ModelAttribute("seccion") Integer pSeccion
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a las funciones asociarCategorias y desasociarCategorias del servicio SeccionService. Sus parámetros serán el código de sección recibido por pantalla y la composición de listas de asociación y desasociación de categorías correspondientes para actualizarlas en la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que las categorías se han actualizado usando el componente alerts de tipo success de Bootstrap.

- Desasociar categoría:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Desasociar categoría" del formulario de edición de la sección seleccionada. Alojada en la clase controlador de Spring AdminSeccionController, su especificación será la siguiente:

```
@RequestMapping(
        value = "desasociarCategoria"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String desasociarCategoria(
        @ModelAttribute("categoria") Integer pCategoria
      , @ModelAttribute("seccion") Integer pSeccion
      , @ModelAttribute("optClicked") Integer pOpClicked
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función desasociarCategorias del servicio SeccionService, con el código de sección recibido por pantalla y la categoría a desasociar, para actualizarla en la base de datos. En caso de que la operación en la base de datos se realice correctamente se mostrará en pantalla que la categoría se ha desasociado usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de secciones se mostrarán en la parte superior de la sección (encima del artículo con los criterios de búsqueda) usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager se redirige a la página de error con un mensaje de falta de permisos.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Comprobar que al dar de alta una sección: el formulario se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que al pulsar el botón “Editar” se despliega el formulario de edición cumplimentado correctamente.
- Modificar los datos de una sección, pulsar “Guardar” y comprobar que los datos se actualizan correctamente en la base de datos. Validar que se muestra el feedback en la pantalla.
- Pulsar “Borrar” y comprobar que la sección se ha eliminado de la base de datos. Validar que ha desaparecido el artículo y que se muestra el feedback en la

pantalla.

- Pulsar en “Editar” para modificar los datos de una sección y después pulsar en el “Editar” de otra. Comprobar que sólo se muestra el formulario de edición de la segunda sección.
- Pulsar en “Editar” para modificar los datos de una sección, realizar modificaciones y después pulsar en el “Editar” de otra. Comprobar que se muestra un diálogo de confirmación para deshacer los cambios de la primera sección antes de continuar.
- Modificar los datos de una sección, pulsar “Cancelar” y comprobar que se cancelan las modificaciones correctamente.
- Pulsar el botón “Publicar” de la sección dada de alta y comprobar que se marca en la base de datos como publicada (PUBLICADO = 'S'). Comprobar también que al desplegar el formulario de edición tiene chequeado el campo “Publicado” y que la fecha es la del día en el campo “Fecha de publicación”. Tiene que mostrarse el botón “Despublicar” en lugar de “Publicar”.

  Validar que se muestra el feedback de la operación en la pantalla.
- Pulsar el botón “Despublicar” de la sección que se acaba de publicar y comprobar que se marca en la base de datos como no publicada (PUBLICADO = 'N'). Comprobar también que al desplegar el formulario de edición no tiene chequeado el campo “Publicado” y que el campo “Fecha de publicación” está vacío. Tiene que mostrarse el botón “Publicar” en lugar de “Despublicar”.

  Validar que se muestra el feedback de la operación en la pantalla.
- Asociar categorías a una sección y comprobar que los cambios se realizan correctamente en la base de datos. Validar que el combo “Categorías” se actualiza correctamente y se muestra el feedback en la pantalla.
- Seleccionar una categoría en el combo “Categorías” y pulsar el botón “Desasociar categoría”. Comprobar que la categoría ya no está asociada a la sección en la base de datos y que el combo “Categorías” se actualiza correctamente.

  Validar que se muestra el feedback de la operación en la pantalla.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 8% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (8% sobre 1000 horas aprox.)	84 horas
Consumidas en diseño	7 horas
Disponibles para implementación	77 horas

Tabla 27: Gestión de secciones - Planificación

3.4.1.9. Gestión de categorías

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de categorías. En el análisis se especifica que un usuario puede:

- Crear categoría.
- Asociar sección.
- Desasociar sección.
- Modificar categoría.
- Eliminar categoría.

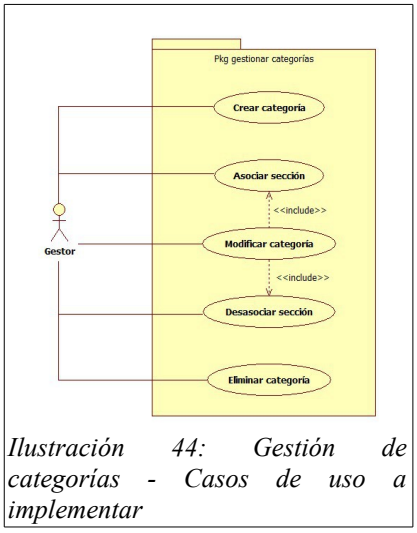


Ilustración 44: Gestión de categorías - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

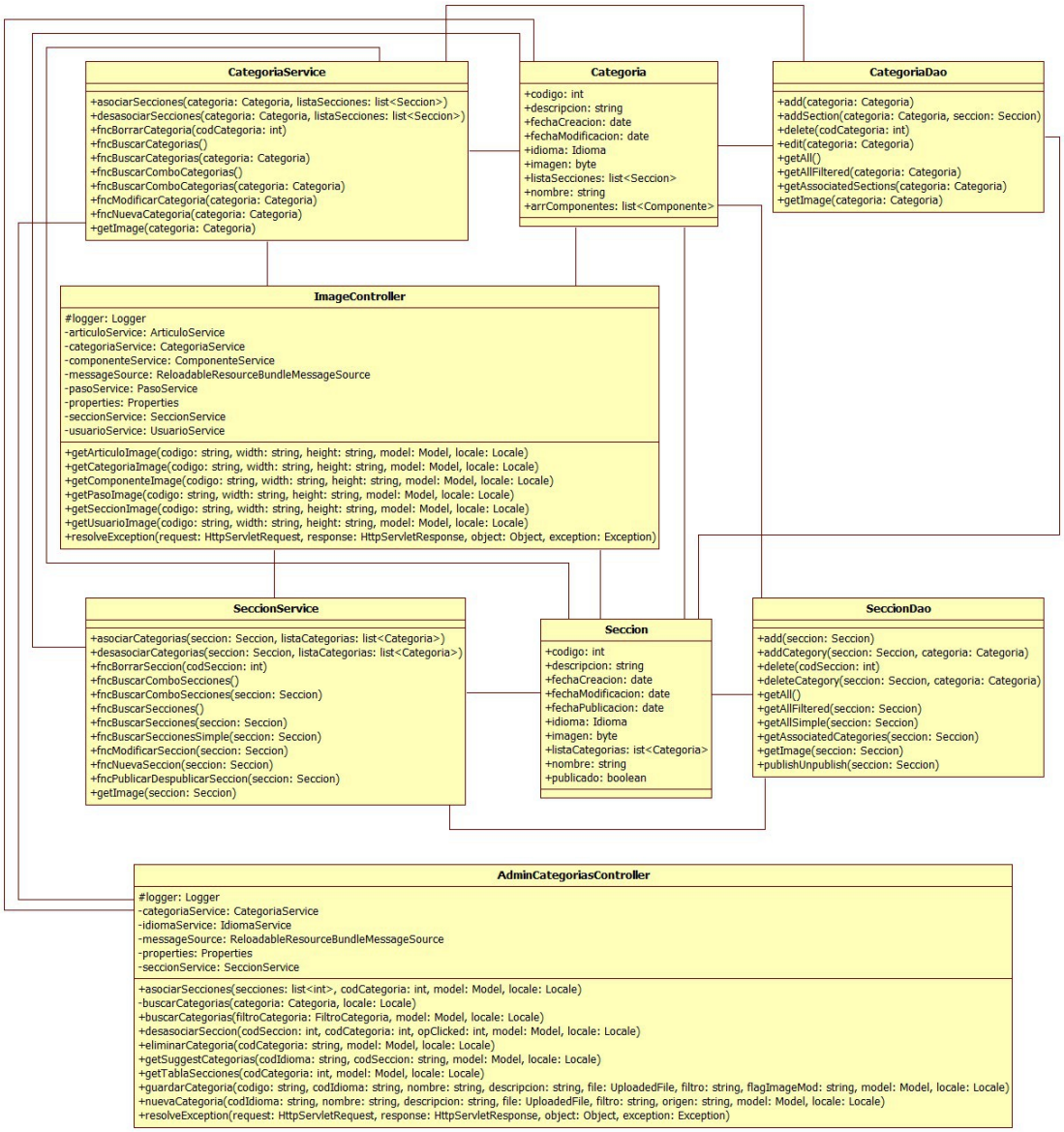


Ilustración 45: Gestión de categorías – Clases empleadas

Entidades empleadas:

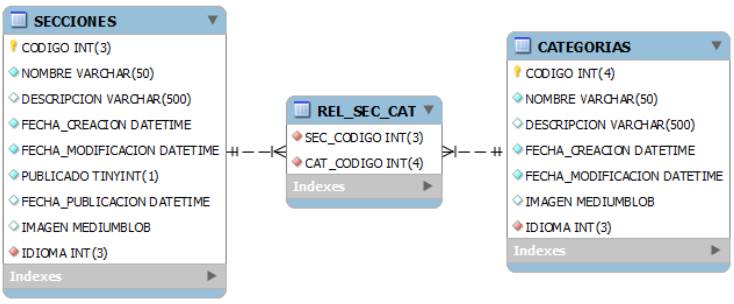


Ilustración 46: Gestión de categorías - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página.

Diseño de la pantalla

La pantalla mostrará varios artículos en la sección. El primero de ellos contendrá el formulario de búsqueda de categorías. Cada uno de los siguientes se corresponderá con una de las categorías encontradas al ejecutar la búsqueda.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Idioma: Combo cargado con los idiomas disponibles.
- Sección: Combo cargado con las secciones dadas de alta en la base de datos. Será dependiente del combo de idioma.
- Categoría: Campo de texto simple para filtrar las categorías por nombre. Se requiere que este campo sugiera categorías que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI.
- Botón de buscar: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-search". Al hacer clic sobre este botón, se enviará el formulario por método POST para buscar categorías que coincidan con el filtro.
- Botón nuevo: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-plus". Al hacer clic sobre este botón, se mostrará un formulario en una ventana modal creada con el componente Bootstrap Dialog. Su título será "Nueva categoría" y contendrá un formulario con los siguientes campos:
  - Idioma: Combo obligatorio con los idiomas disponibles.

- Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
- Descripción: Textarea obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
- Imagen: Campo de tipo fichero.

Cada artículo de resultado de búsqueda mostrará la siguiente información:

- Nombre: El nombre de la categoría.
- Descripción: Descripción de la categoría.
- Imagen: Imagen de la categoría. Será un enlace que al pulsarse cargará la imagen a tamaño completo en la pantalla con el componente BlockUI.
- Menú de opciones: En la parte superior derecha del artículo se mostrará un menú de opciones a realizar con la sección. Las opciones serán las siguientes:
  - Editar:

    Botón por defecto de Bootstrap sin texto y con el glyphicon "glyphicon-pencil".

    Al pulsar este botón se desplegará, debajo de los datos mostrados de la categoría, un formulario de edición de la sección. Este formulario contendrá los siguientes campos:

    - Fecha creación: Campo de texto simple deshabilitado y precargado con la fecha de creación de la categoría.
    - Fecha modificación: Campo de texto simple deshabilitado y precargado con la fecha de la última modificación de la categoría.
    - Idioma: Combo obligatorio con todos los idiomas disponibles precargado con el idioma de la categoría.
    - Nombre: Campo obligatorio de texto simple y precargado con el nombre de la categoría. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Descripción: Textarea obligatorio precargado con la descripción de la categoría. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Imagen: Campo de tipo fichero.
    - Secciones: Combo precargado con las secciones asociadas a la categoría.
    - Asociar sección: Botón por defecto de Bootstrap que tiene un span a la izquierda del texto con el glyphicon "glyphicon-plus-sign". Al hacer clic sobre este botón se mostrará un formulario en una ventana modal creada

con el componente Bootstrap Dialog. Su título será “Asociar sección” y visualiza una búsqueda de secciones. En esta ventana se mostrará:

- Un campo de texto simple para filtrar las secciones por nombre.
- Una tabla creada con el componente FooTable que contendrá todas las secciones dadas de alta en el idioma de la categoría.

  Las filas de la tabla mostrarán un checkbox de selección y el nombre de la sección. Como campo oculto tendremos la descripción de la misma. El campo de selección aparecerá chequeado en caso de que la sección se encuentre asociada a la categoría que se está editando.
- Un botón de Bootstrap de tipo principal con el texto “Asociar”. Al pulsarse se cerrará la ventana y se lanzará una operación de ajax con las secciones seleccionadas para que se guarden en la base de datos. Esta operación de ajax retornará las secciones con sus nombres para actualizar el combo “Secciones” de la categoría.

  En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.
- Un botón por defecto de Bootstrap con el texto “Cancelar”. Al pulsarse se cerrará la ventana.

- Desasociar sección: Desasocia de la categoría editada la sección seleccionada en el combo “Secciones”. Se realizará por medio de una operación de ajax.

  En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.

Al pulsarse el botón de edición, el artículo de la sección pasará a estar en modo edición y el botón “Editar” se ocultará en favor de otros dos botones:

- Guardar: Botón de tipo principal de Bootstrap sin texto y con el glyphicon “glyphicon-floppy-disk”. Al pulsar este botón se comprobará que el formulario de edición sea válido. En caso de ser así se enviará al servidor.
- Cancelar: Botón por defecto de Bootstrap sin texto y con el glyphicon “glyphicon-floppy-remove”. Al pulsar este botón se colapsa el formulario de edición y se restaura, deshaciéndose los cambios.

No se podrán tener dos formularios de edición abiertos. Si se va a abrir un segundo formulario de edición, se cerrará el que esté abierto en ese momento de forma automática. Si se da el caso de que el formulario a cerrar automáticamente tuviera cambios sin guardar se preguntará al usuario si

desea descartarlos antes de continuar o si por lo contrario prefiere cancelar la edición de la segunda sección.

- Borrar:

  Botón de tipo danger de Bootstrap sin texto y con el glyphicon “glyphicon-trash”.

  Por medio de una operación de ajax se realizará el borrado en la base de datos de la categoría y sus relaciones con las secciones.

  En caso de que la operación haya resultado satisfactoria, se mostrará un texto con el feedback encima del artículo de criterios de búsqueda por medio del componente alerts de tipo success de Bootstrap.

Diseño funcional

Las funciones que se ejecutarán en servidor para la gestión de categorías serán las siguientes:

- Búsqueda de categorías:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "buscarCategorias"
      , method = RequestMethod.POST)
   public String buscarCategorias(
        @ModelAttribute("categoriaFiltro") FiltroCategoria categoriaFiltro
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncBuscarCategorias del servicio CategoriaService para obtener los datos de las categorías para el filtro especificado en la pantalla. Una vez metida la lista de categorías en el modelo de Spring, se redireccionará a la pantalla de gestión de categorías y mostrará los resultados.

- Alta de categoría:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar” del formulario de alta de categoría. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevaCategoria"
      , method = RequestMethod.POST)
   public String nuevaCategoria(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevaCategoria del servicio CategoriaService para insertar la categoría con los parámetros recibidos por la pantalla. En caso de que la operación en la base de datos se realice correctamente se mostrará en la pantalla que la categoría se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de categoría:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario edición de la categoría. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarCategoria"
      , method = RequestMethod.POST)
   public String guardarCategoria(
        @ModelAttribute("codigo") String codigo
      , @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("flagImageMod") String flagImageMod
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncModificarCategoria del servicio CategoriaService para actualizar la categoria con los parámetros recibidos por la pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la categoría se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Borrado de categoría:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Borrar" de la categoría seleccionada. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarCategoria"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarCategoria(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncBorrarCategoria del servicio CategoriaService con el código de categoría recibido por la pantalla para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la categoría se ha eliminado usando el componente alerts de tipo success de Bootstrap.

- Asociar secciones:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Asociar sección" del formulario de edición de la categoría seleccionada. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "asociarSecciones"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String asociarSecciones(
        @ModelAttribute("secciones") List<Integer> pSecciones
      , @ModelAttribute("categoria") Integer pCategoria
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a las funciones asociarSecciones y desasociarSecciones del servicio CategoriaService. Sus parámetros serán el código de categoría recibido por pantalla y la composición de listas de asociación y desasociación de las secciones correspondientes para actualizarlas en la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que las secciones se han actualizado usando el componente alerts de tipo success de Bootstrap.

- Desasociar sección:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Desasociar sección" del formulario de edición de la categoría seleccionada. Alojada en la clase controlador de Spring AdminCategoriasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "desasociarSeccion"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String desasociarSeccion(
        @ModelAttribute("seccion") Integer pSeccion
      , @ModelAttribute("categoria") Integer pCategoria
      , @ModelAttribute("optClicked") Integer pOpClicked
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método desasociarSecciones del servicio CategoriaService. Los parámetros serán el código de categoría recibido por pantalla y la sección a desasociar para actualizarla en la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la sección se ha desasociado usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de categorías se mostrarán en la parte superior de la sección (encima del artículo con los criterios de búsqueda) usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación, se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager, se redirige a la página de error con un mensaje de falta de permisos.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Comprobar que al dar de alta una categoría: el formulario se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que al pulsar el botón “Editar”, se despliega el formulario de edición cumplimentado correctamente.
- Modificar los datos de una categoría, pulsar “Guardar” y comprobar que los datos se actualizan correctamente en la base de datos. Validar que se muestra el feedback en la pantalla.
- Pulsar “Borrar” y comprobar que la categoría se ha eliminado de la base de

datos. Validar que ha desaparecido el artículo y que se muestra el feedback en la pantalla.

- Pulsar en “Editar” para modificar los datos de una categoría y después pulsar en el “Editar” de otra. Comprobar que sólo se muestra el formulario de edición de la segunda categoría.
- Pulsar en “Editar” para modificar los datos de una categoría, realizar modificaciones y después pulsar en el “Editar” de otra. Comprobar que se muestra un diálogo de confirmación para deshacer los cambios de la primera categoría antes de continuar.
- Modificar los datos de una categoría, pulsar “Cancelar” y comprobar que se cancelan las modificaciones correctamente.
- Asociar secciones a una categoría y comprobar que los cambios se realizan correctamente en la base de datos. Validar que el combo “Secciones” se actualiza correctamente y se muestra el feedback en la pantalla.
- Seleccionar una sección en el combo “Secciones” y pulsar el botón “Desasociar sección”. Comprobar que la sección ya no está asociada a la categoría en la base de datos y que el combo “Secciones” se actualiza correctamente.

  Validar que se muestra el feedback de la operación en la pantalla.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 4% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (4% sobre 1000 horas aprox.)	42 horas
Consumidas en diseño	3 horas
Disponibles para implementación	39 horas

Tabla 28: Gestión de categorías - Planificación

3.4.1.10. Gestión de componentes

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de componentes. En el análisis se especifica que un usuario puede:

- Crear componente.
- Modificar componente.
- Asociar etiqueta.
- Eliminar componente.

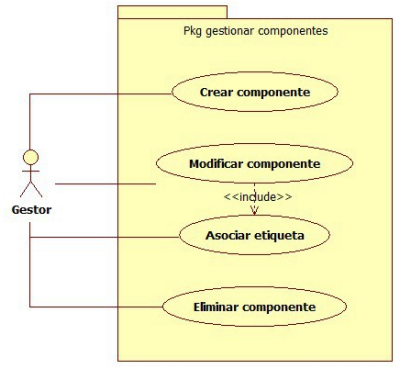


Ilustración 47: Gestión de componentes - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

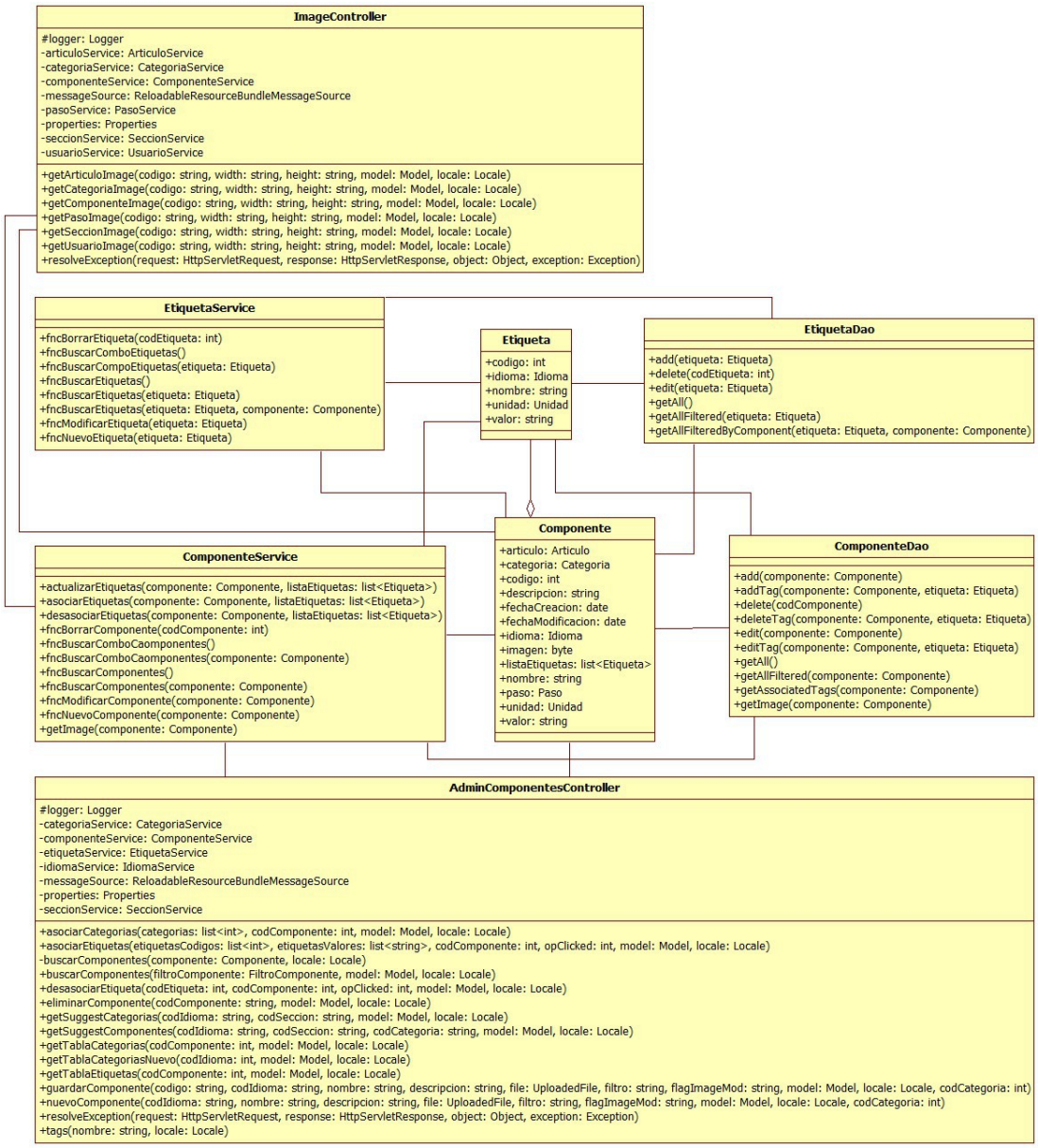


Ilustración 48: Gestión de componentes – Clases empleadas

Entidades empleadas:

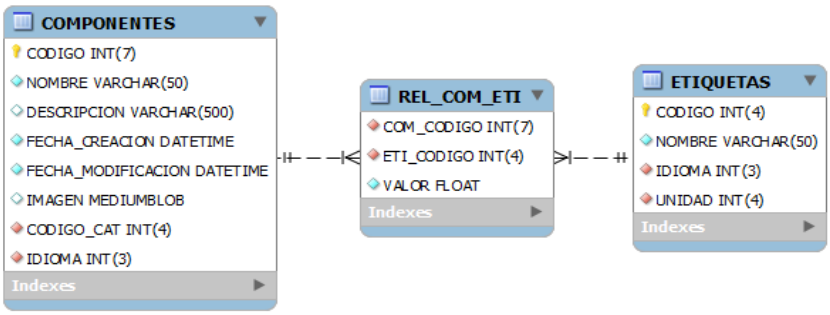


Ilustración 49: Gestión de componentes - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con el perfil de administrador o de mánager tendrán acceso a esta página.

Diseño de la pantalla

La pantalla mostrará dos artículos en la sección. Uno de ellos contendrá el formulario de búsqueda de componentes y el otro la tabla de resultados encontrados.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Idioma: Combo con los idiomas disponibles.
- Sección: Combo con las secciones dadas de alta en la base de datos. El contenido de este combo dependerá del idioma seleccionado en el combo de idioma.
- Categoría: Campo de texto simple para filtrar los componentes por categoría. Se requiere que este campo sugiera categorías que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI. Este campo dependerá del combo de idioma y del combo de sección.
- Componente: Campo de texto simple para filtrar los componentes por nombre. Se requiere que este campo sugiera componentes que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI. Este campo dependerá del combo de idioma, del combo de sección y del campo de categoría.
- Botón de buscar: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-search". Al hacer clic sobre este botón se enviará el formulario por método POST para buscar usuarios con el nombre del filtro.

- Botón nuevo: Botón por defecto de Bootstrap sin texto con el glyphicon “glyphicon-plus”. Al hacer clic sobre este botón se mostrará un formulario en una ventana modal, creada con el componente Bootstrap Dialog. Su título será “Nuevo componente” y contendrá un formulario con los siguientes campos:
    - Idioma: Combo obligatorio con los idiomas disponibles.
    - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Descripción: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Imagen: Campo de tipo fichero.
    - Categoría: Campo de texto simple deshabilitado con la categoría asociada. Este campo es obligatorio.
    - Asociar categoría: Botón por defecto de Bootstrap que muestra una ventana modal creada con el componente Bootstrap Dialog. Esta nueva ventana visualiza una búsqueda de categorías. Aquí se mostrará:
        - Un campo de texto simple para filtrar las categorías por nombre.
        - Una tabla creada con el componente FooTable que contendrá todas las categorías dadas de alta.

          Las filas de la tabla mostrarán un radio de selección y el nombre de la categoría. Como campo oculto tendremos la descripción de la misma.
        - Un botón de Bootstrap de tipo principal con el texto “Asociar”. Al pulsarse se cerrará la ventana. En caso de haberse seleccionado alguna categoría, ésta aparecerá en el campo de texto de categoría.
        - Un botón por defecto de Bootstrap con el texto “Cancelar”. Al pulsarse se cerrará la ventana.
    - Guardar: Botón de tipo principal de Bootstrap que al pulsarlo valida el formulario. En caso de ser correcto lo envía.
    - Cancelar: Botón por defecto de Bootstrap que al pulsarlo cancela el alta y cierra la ventana modal.

En el segundo artículo se enseña la tabla de resultados formada con el componente FooTable. Se visualizan los siguientes campos en sus diferentes configuraciones:

- Escritorio: Imagen, nombre, descripción y categoría.
- Tableta: Imagen, nombre, descripción y categoría.
- Móvil: Nombre y categoría.
- Siempre oculto: Botones de etiquetas, editar y borrar.

Se ordenará por defecto por la columna de nombre de forma ascendente. Será posible la ordenación por cualquier columna salvo por la de imagen.

Las imágenes de componentes serán enlaces que mostrarán la foto del componente en tamaño completo mediante el componente BlockUI.

Al pulsar en el botón de “Etiquetas” se desplegará de la fila un formulario con los siguientes datos:

- Tabla:

  Tabla creada con el componente FooTable que contendrá las etiquetas asociadas al componente. La tabla siempre mostrará tres columnas:

  - La primera será un radio para seleccionar la fila.
  - La segunda tendrá el nombre de la etiquetas.
  - La tercera llevará el valor de la etiqueta en el componente.

  La tabla podrá ordenarse por el nombre y por el valor.

- Asociar Etiqueta:

  Botón por defecto de Bootstrap que muestra una segunda ventana modal creada con el componente Bootstrap Dialog que muestra una búsqueda de etiquetas. En esta ventana se mostrará:

  - Un campo de texto simple para filtrar las etiquetas por el nombre.
  - Una tabla creada con el componente FooTable que contendrá todas las etiquetas dadas de alta para el idioma del componente.

    En la tabla se mostrarán una columna “Nombre” con el nombre de la etiqueta y otra columna “Valor” que contendrá por cada fila un campo de texto simple donde insertar el valor asociado al componente. Para que una etiqueta esté asociada a un componente, requiere que tenga un valor específico.
  - Un botón de Bootstrap de tipo principal con el texto “Asociar”. Al pulsarse se cerrará la ventana y se actualizarán en la base de datos las etiquetas modificadas.
  - Un botón por defecto de Bootstrap con el texto “Cancelar”. Al pulsarse, se cerrará la ventana.

Al pulsar en el botón “Editar” se desplegará de la fila un formulario con los siguientes datos:

- Fecha creación:

  Campo de texto simple deshabilitado con la fecha de alta del componente precargado.

- Fecha modificación:

  Campo de texto simple deshabilitado con la fecha de la última modificación del componente precargado.

- Nombre:

  Campo de texto obligatorio. Estará cargado con el nombre del componente. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Descripción:

  Textarea obligatorio. Estará cargado con la descripción del componente. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Imagen:

  Input de tipo file si el usuario no tenía imagen guardada o un botón por defecto de Bootstrap con el texto “Cambiar imagen”. Al pulsar en este botón se ocultará y aparecerá un input de tipo file para añadir la nueva imagen. No es obligatorio, con lo que si modificamos y no añadimos una imagen nueva borraremos la que ya estaba guardada en la base de datos.

- Categoría: Campo de texto simple deshabilitado con la categoría asociada. Este campo es obligatorio.

- Asociar categoría: Botón por defecto de Bootstrap que muestra una ventana modal creada con el componente Bootstrap Dialog con una búsqueda de categorías. Esta ventana contendrá:

  - Un campo de texto simple para filtrar las categorías por nombre.

  - Una tabla creada con el componente FooTable que contendrá todas las categorías dadas de alta.

    Las filas de la tabla mostrarán un radio de selección y el nombre de la categoría. Como campo oculto tendremos la descripción de la misma.

  - Un botón de Bootstrap de tipo principal con el texto “Asociar”. Al pulsarse se cerrará la ventana. En caso de haberse seleccionado alguna categoría, ésta aparecerá en el campo de texto de categoría.

  - Un botón por defecto de Bootstrap con el texto “Cancelar”. Al pulsarse se cerrará la ventana.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente. En caso satisfactorio lanzará la acción de modificar componente.

- Botón de cancelar:

  Será un botón de tamaño norma tipo danger de Bootstrap. Cancela las modificaciones realizadas en el componente y oculta el formulario de edición.

No se podrán tener dos formularios de edición abiertos. Si se va a abrir un segundo formulario de edición se cerrará el que esté abierto en ese momento de forma automática. Si se da el caso de que el formulario a cerrar automáticamente tenía cambios sin guardar, se preguntará al administrador si desea descartarlos antes de continuar o por lo contrario prefiere cancelar la edición del segundo componente.

Al pulsar el botón “Borrar” se llamará por ajax a la función de borrado de componentes. Si el borrado se ha realizado correctamente, se elimina la fila de la tabla con el componente.

Diseño funcional

Las funciones que se ejecutarán en servidor para la gestión de componentes serán las siguientes:

- Búsqueda de componentes:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "buscarComponentes"
      , method = RequestMethod.POST)
   public String buscarComponentes(
        @ModelAttribute("componenteFiltro") FiltroComponente componenteFiltro
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar a la función fncBuscarComponentes del servicio ComponenteService para obtener los datos de los componentes que coincidan con el filtro de la pantalla. Una vez metida la lista de usuarios en el modelo de Spring, redireccionará a la pantalla de gestión de componentes y mostrará los resultados.

- Alta de componente:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar” del formulario de alta de componente. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevoComponente"
      , method = RequestMethod.POST)
   public String nuevoComponente(
        @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , @ModelAttribute("searched") String searched
      , @ModelAttribute("codCategoria") Integer codCategoria
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncNuevoComponente, del servicio ComponenteService, para insertar el componente con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en pantalla que el componente se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de componente:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario edición del componente. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarComponente"
      , method = RequestMethod.POST)
   public String guardarComponente(
        @ModelAttribute("codigo") String codigo
      , @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("FORM") UploadedFile file
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("flagImageMod") String flagImageMod
      , @ModelAttribute("codCategoria") Integer codCategoria
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncModificarComponente, del servicio ComponenteService, para actualizar el componente con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el componente se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Borrado de componente:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Borrar" de la fila de componente seleccionada. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarComponente"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarComponente(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncBorrarComponente, del servicio ComponenteService, con el código de componente recibido por pantalla para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el componente se ha eliminado usando el componente alerts de tipo success de Bootstrap.

- Asociar categoría:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Asociar" de la ventana modal de "Asociar categoría". Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "asociarCategorias"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String asociarCategorias(
        @ModelAttribute("categorias") List<Integer> pCategorias
      , @ModelAttribute("componente") Integer pComponente
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a la función fncModificarComponente, del servicio ComponenteService, con el código de componente recibido por pantalla para asociarle la categoría especificada. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la categoría se ha asociado usando el componente alerts de tipo success de Bootstrap.

- Asociar etiqueta:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Asociar etiqueta" de la fila de componente asociada. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
        value = "asociarEtiquetas"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String asociarEtiquetas(
        @ModelAttribute("etiquetasCodigos") List<Integer> pEtiquetasCodigos
      , @ModelAttribute("etiquetasValores") List<String> pEtiquetasValores
      , @ModelAttribute("componente") Integer pComponente
      , @ModelAttribute("opClicked") Integer opClicked
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar a las funciones: actualizarEtiquetas, asociarEtiquetas y desasociarEtiquetas del servicio ComponenteService. Los parámetros serán el código de componente recibido por pantalla y las listas compuestas con las etiquetas y sus valores para asociar o desasociar según convenga. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que las etiquetas se han actualizado usando el componente alerts de tipo success de Bootstrap.

- Desasociar etiqueta:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Desasociar etiqueta" de la fila de componente asociada. Alojada en la clase controlador de Spring AdminComponentesController, su especificación será la siguiente:

```
@RequestMapping(
                value = "desasociarEtiqueta"
              , method = RequestMethod.POST
              , produces = "application/x-json;charset=UTF-8")
       public @ResponseBody String desasociarEtiqueta(
                @ModelAttribute("etiqueta") Integer pEtiqueta
              , @ModelAttribute("componente") Integer pComponente
              , @ModelAttribute("optClicked") Integer pOpClicked
              , Model model, Locale locale) {
       //TODO
}
```

Esta función deberá llamar a la función desasociarEtiquetas del servicio ComponenteService, con el código de componente recibido por pantalla y la etiqueta seleccionada. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la etiqueta se ha desasociado usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de componentes se mostrarán en la parte superior de la sección (encima del artículo con los criterios de búsqueda) usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager se redirige a la página de error con un mensaje de falta de permisos.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Redimensionar la pantalla y comprobar que el componente de tabla cambia correctamente.
- Comprobar que al dar de alta un componente el formulario: se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que al pulsar el botón “Etiquetas”, se despliega la capa de etiquetas asociadas al componente con sus valores correctamente.
- Comprobar que: las etiquetas a asociar y desasociar se guardan correctamente en la base de datos, la tabla de etiquetas se actualiza correctamente y se muestra el feedback con las operaciones realizadas.
- Comprobar que al pulsar el botón “Editar” se despliega el formulario de edición cumplimentado correctamente.
- Comprobar que al pulsar el botón “Cambiar imagen” éste desaparece y se muestra el campo para insertar imagen.
- Modificar los datos de un componente, pulsar “Guardar” y comprobar que los datos se actualizan correctamente en la base de datos. Validar que se muestra el feedback en la pantalla.
- Pulsar “Borrar” y comprobar que el componente se ha eliminado de la base de datos. Validar que ha desaparecido de la tabla y que se muestra el feedback en la pantalla.
- Pulsar en “Editar” para modificar los datos de un componente y después pulsar en el “Editar” de otro. Comprobar que sólo se muestra el formulario de edición del segundo componente.
- Pulsar en “Editar” para modificar los datos de un componente, realizar modificaciones y después pulsar en el “Editar” de otro componente. Comprobar que se muestra un diálogo de confirmación para deshacer los cambios del primero antes de continuar.
- Modificar los datos de un componente, pulsar “Cancelar” y comprobar que se

cancelan las modificaciones correctamente.

- Comprobar al asociar una categoría ésta se guarda en la base de datos correctamente y que muestra el feedback en la pantalla.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 17% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (17% sobre 1000 horas aprox.)	168 horas
Consumidas en diseño	8 horas
Disponibles para implementación	160 horas

Tabla 29: Gestión de componentes - Planificación

3.4.1.11. Gestión de etiquetas

Caso o casos de uso a implementar.

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de etiquetas. En el análisis se especifica que un usuario puede:

- Crear etiqueta.
- Modificar etiqueta.
- Eliminar etiqueta.

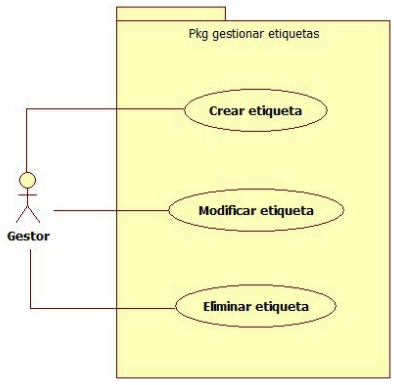


Ilustración 50: Gestión de etiquetas - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

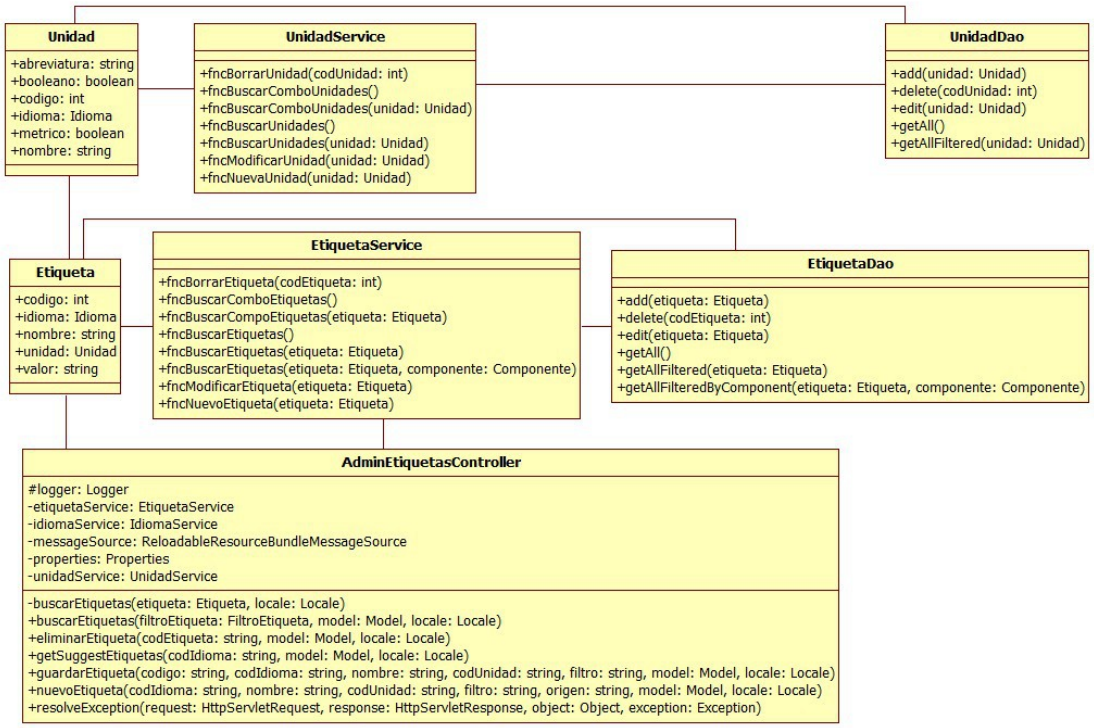


Ilustración 51: Gestión de etiquetas – Clases empleadas

Entidades empleadas:

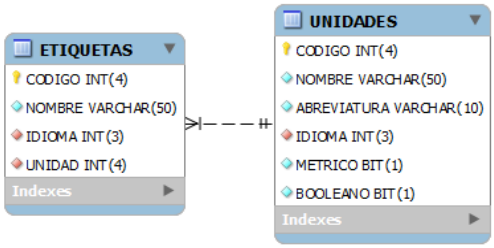


Ilustración 52: Gestión de etiquetas - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página.

Diseño de la pantalla

La pantalla mostrará dos artículos en la sección. Uno de ellos contendrá el formulario de búsqueda de etiquetas y el otro la tabla de resultados encontrados.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Idioma: Combo con los idiomas disponibles.
- Etiqueta: Campo de texto simple para filtrar las etiquetas por nombre. Se requiere que este campo sugiera etiquetas que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI. Este campo dependerá del combo de idioma.
- Botón de buscar: Botón por defecto de Bootstrap, sin texto, con el glyphicon "glyphicon-search". Al hacer clic sobre este botón, se enviará el formulario por método POST para buscar etiquetas que coincidan con el filtro.
- Botón nuevo: Botón por defecto de Bootstrap, sin texto, con el glyphicon "glyphicon-plus". Al hacer clic sobre este botón se mostrará un formulario en una ventana modal, creada con el componente Bootstrap Dialog, cuyo título será "Nueva etiqueta". El formulario contendrá los siguientes campos:
    - Idioma: Combo obligatorio con los idiomas disponibles.
    - Nombre: Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
    - Unidad: Combo obligatorio con las unidades disponibles. Su contenido dependerá de lo seleccionado en el combo de idioma.
    - Botón de guardar:

      Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente y en ese caso lanzará la acción de guardar una nueva etiqueta.
    - Botón de cancelar:

      Será un botón de tamaño normal por defecto de Bootstrap. Cancela el alta de etiqueta cerrando el diálogo con el formulario.

En el segundo artículo se muestra la tabla de resultados. Está formada con el componente FooTable y contiene los siguientes campos visibles en sus diferentes configuraciones:

- Escritorio: Nombre y unidad.
- Tableta: Nombre y unidad.
- Móvil: Nombre y unidad.

- Siempre oculto: Botones de editar y borrar.

Se ordenará por defecto por la columna de nombre de forma ascendente siendo posible la ordenación por cualquier columna.

Al pulsar en el botón “Editar” se desplegará de la fila un formulario con los siguientes datos:

- Idioma: Combo obligatorio con los idiomas disponibles. El idioma de la etiqueta se encontrará precargado.
- Nombre: Campo de texto obligatorio con el nombre de la etiqueta precargado. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.
- Unidad: Combo obligatorio con las unidades disponibles. Su contenido dependerá de lo seleccionado en el combo de idioma. Vendrá precargado con la unidad de la etiqueta.
- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente y en ese caso lanzará la acción de modificar etiqueta.
- Botón de cancelar:

  Será un botón de tamaño normal tipo danger de Bootstrap. Cancela las modificaciones realizadas en la etiqueta y oculta el formulario de edición.

No se podrán tener dos formularios de edición abiertos. Si se va a abrir un segundo formulario de edición se cerrará el que esté abierto en ese momento de forma automática. Si se da el caso de que el formulario a cerrar tenía cambios sin guardar se preguntará al usuario si desea descartarlos antes de continuar o si por lo contrario prefiere cancelar la edición de la segunda etiqueta.

Al pulsar el botón “Borrar” se llamará por ajax a la función de borrado de etiquetas. Si el borrado se ha realizado se elimina la fila de la tabla con la etiqueta.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la gestión de etiquetas serán las siguientes:

- Búsqueda de etiquetas:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminEtiquetasController, su especificación será la siguiente:

```
@RequestMapping(
      value = "buscarEtiquetas"
      , method = RequestMethod.POST)
   public String buscarEtiquetas(
        @ModelAttribute("etiquetaFiltro") FiltroEtiqueta etiquetaFiltro
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncBuscarEtiquetas, del servicio EtiquetaService, para obtener los datos de los usuarios para el filtro de la pantalla. Una vez metida la lista de etiquetas en el modelo de Spring se redireccionará a la pantalla de gestión de etiquetas y mostrará los resultados.

- Alta de etiqueta:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario de alta de etiqueta. Alojada en la clase controlador de Spring AdminEtiquetasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevoEtiqueta"
      , method = RequestMethod.POST)
   public String nuevoEtiqueta(
      @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("unidad") String unidad
      , @ModelAttribute("filtro") String filtro
      , @ModelAttribute("origen") String origen
      , @ModelAttribute("searched") String searched
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncNuevoEtiqueta, del servicio EtiquetaService, para insertar la etiqueta con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la etiqueta se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de etiqueta:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Guardar" del formulario edición de la etiqueta. Alojada en la clase controlador de Spring AdminEtiquetasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarEtiqueta"
      , method = RequestMethod.POST)
   public String guardarEtiqueta(
        @ModelAttribute("codigo") String codigo
      , @ModelAttribute("idioma") String idioma
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("unidad") String unidad
      , @ModelAttribute("filtro") String filtro
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncModificarEtiqueta, del servicio EtiquetaService, para actualizar la etiqueta con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que la etiqueta se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Borrado de etiqueta:

  Esta función deberá ser llamada cuando se haga clic sobre el botón "Borrar" de la fila de etiqueta seleccionada. Alojada en la clase controlador de Spring AdminEtiquetasController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarEtiqueta"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarEtiqueta(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncBorrarEtiqueta, del servicio EtiquetaService, con el código de etiqueta recibido por pantalla para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en pantalla que la etiqueta se ha eliminado usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de etiquetas se mostrarán en la parte superior de la sección (encima del artículo con los criterios de búsqueda) usando el componente alerts de tipo danger de Bootstrap.

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager se redirige a la página de error con un mensaje de falta de permisos.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Comprobar que al dar de alta una etiqueta el formulario: se valida correctamente, los datos se guardan bien en la base de datos y aparece el mensaje de confirmación en la pantalla.
- Comprobar que al pulsar el botón “Editar” se despliega el formulario de edición cumplimentado correctamente.
- Modificar los datos de una etiqueta, pulsar “Guardar” y comprobar que los datos se actualizan correctamente en la base de datos. Validar que se muestra el feedback en la pantalla.
- Pulsar “Borrar” y comprobar que la etiqueta se ha eliminado de la base de datos. Validar que ha desaparecido de la tabla y que se muestra el feedback en la pantalla.
- Pulsar en “Editar” para modificar los datos de una etiqueta y después pulsar en el “Editar” de otra. Comprobar que sólo se muestra el formulario de edición de la segunda etiqueta.
- Pulsar en “Editar” para modificar los datos de una etiqueta, realizar modificaciones y después pulsar en el “Editar” de otra. Comprobar que se muestra un diálogo de confirmación para deshacer los cambios de la primera etiqueta antes de continuar.
- Modificar los datos de una etiqueta, pulsar “Cancelar” y comprobar que se cancelan las modificaciones.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 7% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (7% sobre 1000 horas aprox.)	67 horas
Consumidas en diseño	7 horas
Disponibles para implementación	60 horas

Tabla 30: Gestión de etiquetas - Planificación

3.4.1.12. Gestión de tutoriales

Caso o casos de uso a implementar

Los casos de uso a implementar en la pantalla serán los referentes a la gestión de tutoriales. En el análisis se especifica que un usuario puede:

- Crear tutorial.
- Crear paso.
- Modificar paso.
- Eliminar paso.
- Modificar tutorial.
- Eliminar tutorial.

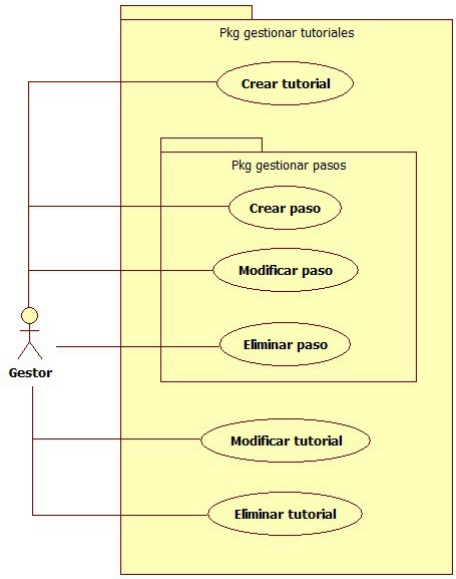


Ilustración 53: Gestión de tutoriales - Casos de uso a implementar

Objetos del modelo y entidades de base de datos

Clases empleadas:

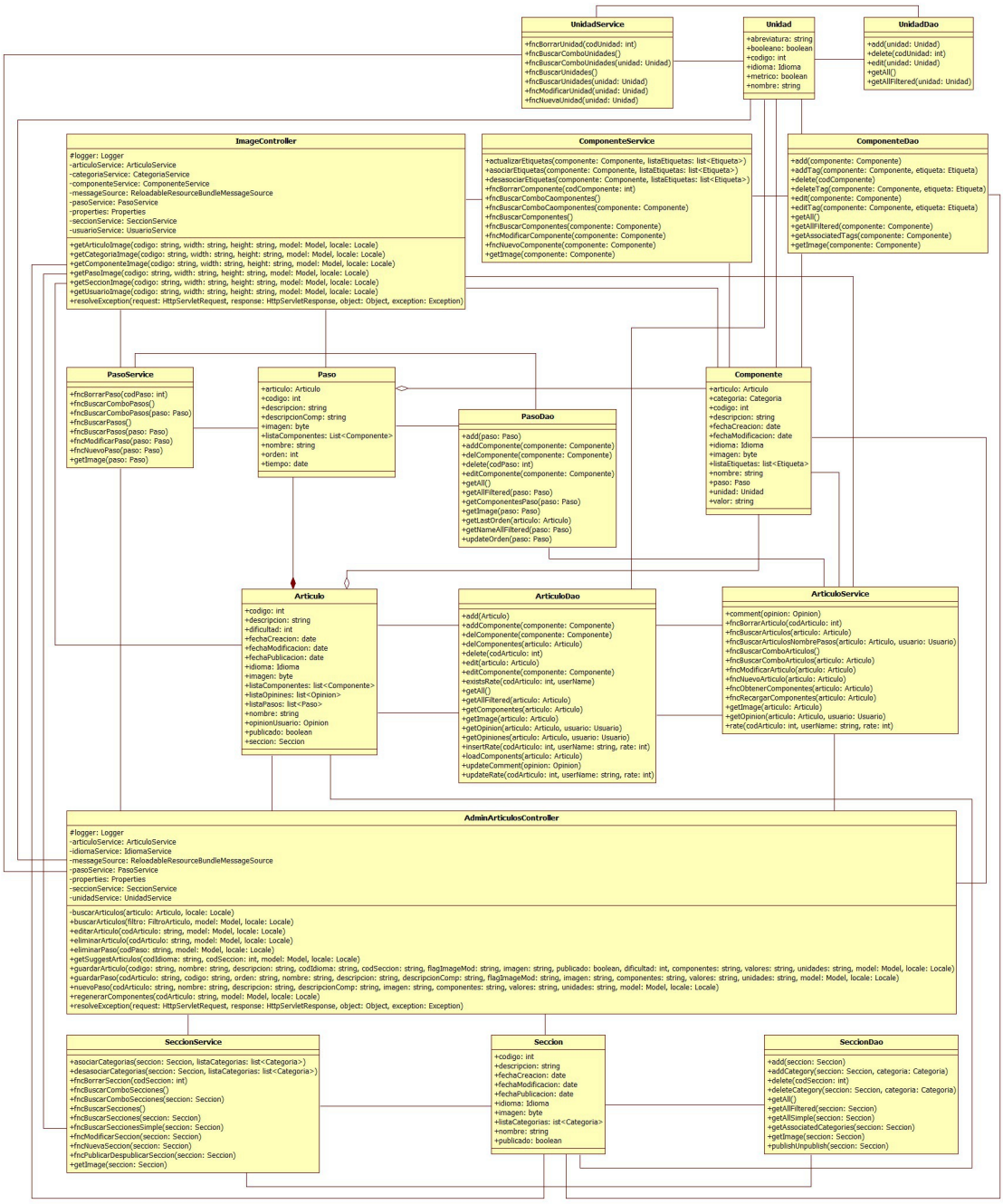


Ilustración 54: Gestión de tutoriales – Clases empleadas

Entidades empleadas:

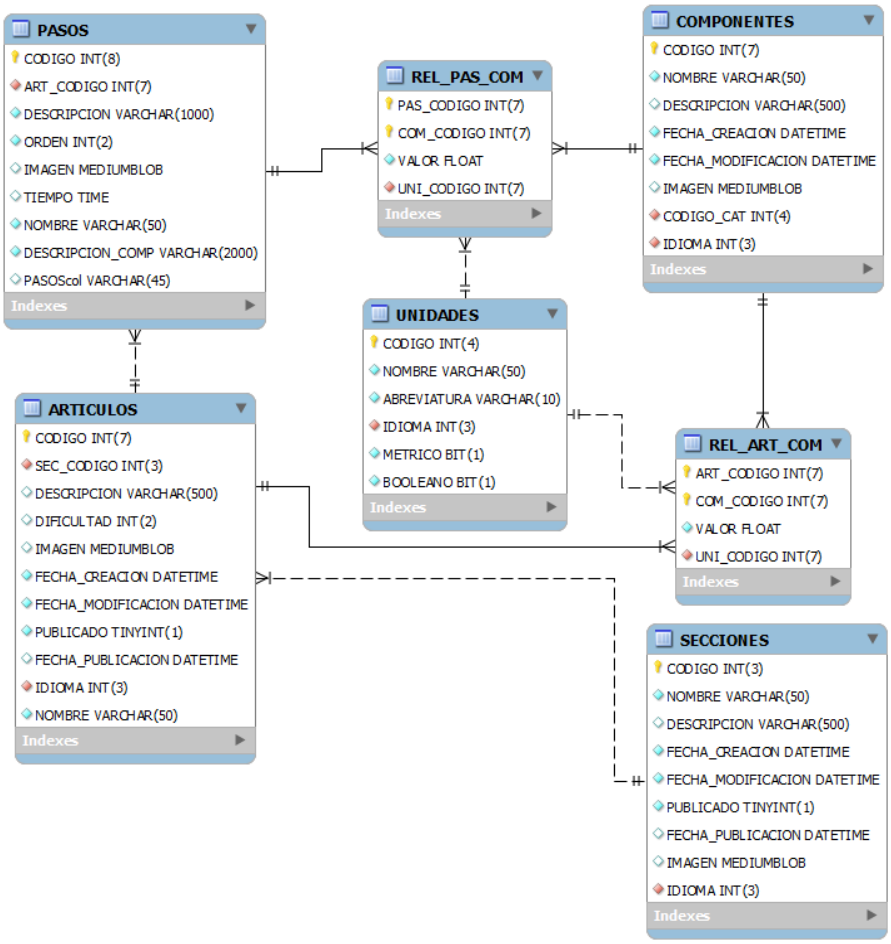


Ilustración 55: Gestión de tutoriales - Entidades empleadas

Implementación de la seguridad

Únicamente los usuarios registrados con perfil de administrador o mánager tendrán acceso a esta página.

Diseño de la pantalla

La pantalla se dividirá en dos, no pudiendo visualizarse las dos a la vez. La primera de ellas corresponde a la pantalla de búsqueda de tutoriales y la segunda al alta/modificación de los mismos.

La pantalla de búsqueda:

Mostrará dos artículos en la sección. Uno de ellos contendrá el formulario de búsqueda de tutoriales y el otro la tabla de resultados encontrados.

El formulario de filtrado irá embebido dentro de un fieldset que tendrá por leyenda "Criterios de búsqueda". Contendrá los siguientes campos:

- Idioma: Combo con los idiomas disponibles.
- Sección: Combo con las secciones disponibles. Su contenido dependerá del idioma seleccionado en el combo de idioma.
- Artículo: Campo de texto simple para filtrar los tutoriales por nombre. Se requiere que este campo sugiera tutoriales que coincidan con el texto que el usuario vaya introduciendo. Para esto se empleará el plugin autocomplete de jQueryUI. Este campo dependerá del combo de idioma y del de sección.
- Botón de buscar: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-search". Al hacer clic sobre este botón se enviará el formulario por método POST para buscar tutoriales que coincidan con el filtro.
- Botón nuevo: Botón por defecto de Bootstrap sin texto con el glyphicon "glyphicon-plus". Al hacer clic sobre este botón se navegará a la pantalla de alta/modificación de tutoriales en modo alta.

En el segundo artículo se mostrará la tabla de resultados formada con el componente FooTable. Contendrá los siguientes campos visibles en sus diferentes configuraciones:

- Escritorio: Imagen, nombre y descripción.
- Tableta: Imagen, nombre y descripción.
- Móvil: Nombre.
- Siempre oculto: Botones de editar y borrar.

Se ordenará por defecto por la columna de nombre de forma ascendente. Será posible la ordenación por cualquier columna excepto la de imagen.

Al pulsar en el botón "Editar" se navegará a la pantalla de alta/modificación de tutoriales en modo modificación.

Al pulsar el botón "Borrar" se llamará por ajax a la función de borrado de tutoriales. Si el borrado se ha realizado correctamente, se eliminará la fila de la tabla con el tutorial.

La pantalla de alta/modificación:

Esta pantalla se divide en tres partes mediante el componente tabs de Bootstrap (http://getbootstrap.com/javascript/#tabs) que da la posibilidad de organizarla en pestañas como si de un archivador se tratara. A continuación se detallan cada una de ellas en orden de aparición:

- Artículo:

  Mostrará la imagen del artículo (si entramos en modo de modificación) y un formulario con los datos básicos del tutorial. El contenido del formulario es el siguiente:

  - Nombre:

    Campo de texto obligatorio con el nombre del tutorial precargado en caso de estar en modo modificación. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

  - Descripción:

    Textarea obligatorio con la descripción del tutorial precargado en caso de estar en modo modificación. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

  - Idioma:

    Combo obligatorio con los idiomas disponibles. Se encontrará precargado con el idioma del tutorial en caso estar en modo modificación.

  - Sección:

    Combo obligatorio con las secciones disponibles. Dependerá del idioma seleccionado en el combo de idiomas. Estará precargado con la sección del tutorial en caso de estar en modo modificación.

  - Dificultad:

    Campo creado con el componente Bootstrap Slider con un rango del 1 al 10. Estará precargado con la dificultad del tutorial en caso de estar en modo modificación.

  - Fecha creación:

    Campo de texto simple deshabilitado. Estará precargado con la fecha de alta del tutorial en caso de estar en modo modificación.

  - Fecha modificación:

    Campo de texto simple deshabilitado. Estará precargado con la fecha de la última modificación del tutorial en caso de estar en modo modificación.

  - Publicado:

    Campo de tipo checkbox que chequeado tomará el valor “S”. En modo modificación estará chequeado si la sección se encuentra publicada. En modo alta por defecto, no se encuentra chequeado.

  - Fecha publicación:

    Campo de texto simple deshabilitado. Estará precargado con la fecha de

publicación del tutorial en caso de estar en modo modificación de un tutorial publicado.

- Imagen:

  Input de tipo file si se está en modo modificación de un tutorial sin imagen o en modo alta. De lo contrario será un botón por defecto de Bootstrap con el texto “Cambiar imagen”. Al pulsar en este botón se ocultará y aparecerá un input de tipo file para añadir la nueva imagen. No es obligatorio, con lo que si se modifica y no se añade una imagen nueva borraremos la que ya estaba guardada en la base de datos.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario haya sido rellenado correctamente. En caso satisfactorio lanzará la acción de modificar tutorial en caso de estar en modo modificación el la de guardar nuevo tutorial en caso contrario.

- Botón de cancelar:

  Será un botón de tamaño normal tipo danger de Bootstrap. Cancela las modificaciones realizadas en el tutorial recargando la pantalla con los mismos datos que cuando se entró por primera vez.

- Botón atrás:

  Será un botón por defecto de Bootstrap que nos devolverá a la pantalla de búsqueda. En caso de haber modificaciones realizadas en el tutorial que no hayan sido guardadas, mostrará un diálogo para confirmar su descarte antes de salir.

- Pasos:

  Todas las acciones que vayan a realizarse en los pasos de un tutorial requerirán un guardado previo de los datos modificados en la primera pestaña.

  En esta pestaña se mostrarán los siguientes datos:

  - Tabla creada con el componente FooTable que: contendrá los pasos del tutorial en el caso de estar en modo modificación de un tutorial con pasos o estará vacía en caso de estar en modo alta.

    La tabla contará con las siguientes columnas visibles en sus diferentes configuraciones:

    - Escritorio: Imagen, nombre y descripción.
    - Tableta: Imagen, nombre y descripción.
    - Móvil: Nombre.
    - Siempre oculto: Botones de editar y borrar.

No se permitirá la ordenación en esta tabla, puesto que los pasos aparecerán ordenados según su orden especificado en la base de datos.

Al pulsar en el botón “Editar” se desplegará el formulario de edición del paso. Este formulario se dividirá en dos partes mediante el componente Bootstrap Wizard (https://github.com/VinceG/twitter-bootstrap-wizard). No será posible navegar de la primera parte del wizard a la segunda sin guardar los datos del paso de la primera.

En la primera parte del wizard se mostrarán los siguientes campos:

- Nombre:

  Campo de texto obligatorio con el nombre del paso precargado. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

- Descripción:

  Textarea obligatorio con la descripción del paso precargado. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

  Este textarea se formará con el componente jQuery Mentions Input (http://podio.github.io/jquery-mentions-input/). Este componente nos permitirá asociar componentes en el paso escribiéndolos directamente en la descripción precedidos por el carácter “#”.

- Imagen:

  Input de tipo file si el paso no tiene imagen. De lo contrario, será un botón por defecto de Bootstrap con el texto “Cambiar imagen”. Al pulsar en este botón se ocultará y aparecerá un input de tipo file para añadir la nueva imagen. No es obligatorio, con lo que si modificamos y no añadimos una imagen nueva borraremos la que ya estaba guardada en la base de datos.

En la segunda parte del wizard se mostrará una tabla con el componente FooTable. Esta tabla almacenará los componentes contenidos en la descripción del paso y sus cantidades necesarias para ejecutarlo.

La tabla de componentes contará con las siguientes columnas visibles en sus diferentes configuraciones:

- Escritorio: Nombre, valor y unidad.
- Tableta: Nombre, valor y unidad.
- Móvil: Nombre.
- Siempre oculto: Imagen.

Únicamente se permitirá la ordenación por nombre en esta tabla.

En la columna valor se presentará un campo de texto simple obligatorio. En

la de unidad se mostrará un combo obligatorio con las unidades disponibles en el idioma del tutorial.

El formulario con el wizard tendrá dos botones:

- Botón guardar: Será un botón de tipo principal de Bootstrap. Al pulsarse se lanzará la acción de modificación de paso.
- Botón cancelar: Omitirá los cambios realizados en el wizard y ocultará el mismo.

Al pulsar el botón "Borrar" se llamará por ajax a la función de borrado de pasos. Si el borrado se ha realizado correctamente, se elimina la fila de la tabla con el paso.

- Nuevo paso:

  Botón por defecto de Bootstrap que abre una ventana modal, creada con el componente Bootstrap Dialog, con el formulario de alta del paso. Este formulario se dividirá en dos partes mediante el componente Bootstrap Wizard. No será posible navegar de la primera parte del wizard a la segunda sin guardar los datos del paso de la primera.

  En la primera parte del wizard se mostrarán los siguientes campos:

  - Nombre:

    Campo de texto obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

  - Descripción:

    Textarea obligatorio. Mostrará el tamaño del campo con el componente Bootstrap MaxLength.

    Este textarea se formará con el componente jQuery Mentions Input. Este componente nos permitirá asociar componentes en el paso directamente, escribiéndolos en la descripción precedidos por el carácter "#".

  - Imagen:

    Input de tipo fichero para la imagen del paso.

  En la segunda parte del wizard se mostrará una tabla con el componente FooTable. Esta tabla almacenará los componentes contenidos en la descripción del paso y sus cantidades necesarias para ejecutarlo.

  La tabla de componentes contará con las siguientes columnas visibles en sus diferentes configuraciones:

  - Escritorio: Nombre, valor y unidad.
  - Tableta: Nombre, valor y unidad.

        - Móvil: Nombre.
        - Siempre oculto: Imagen.

      Únicamente se permitirá la ordenación por nombre en esta tabla.

      En la columna valor se presentará un campo de texto simple obligatorio. En la de unidad se mostrará un combo obligatorio con las unidades disponibles en el idioma del tutorial.

      El formulario con el wizard tendrá dos botones:

        - Botón guardar: Será un botón de tipo principal de Bootstrap. Al pulsarse se lanzará la acción de alta de paso.
        - Botón cancelar: Botón por defecto de Bootstrap que cerrará la ventana modal de alta de paso.

    - Botón de guardar:

      Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario de la primera pestaña haya sido rellenado correctamente. En caso satisfactorio lanzará la acción de modificar tutorial en caso de estar en modo modificación o la de guardar nuevo tutorial en caso contrario.

    - Botón de cancelar:

      Será un botón de tamaño normal tipo danger de Bootstrap. Cancela las modificaciones realizadas en el tutorial en la primera pestaña. Se recarga la pantalla con los mismos datos que cuando se entró por primera vez.

- Componentes:

  En esta pestaña se mostrarán los siguientes datos:

    - Una tabla creada con el componente FooTable. Esta tabla almacenará los componentes empleados en todo el tutorial y sus cantidades necesarias para su consecución en caso de estar en modo modificación de un tutorial con componentes generados. En cualquier otro caso esta tabla estará vacía.

      La tabla de componentes contará con las siguientes columnas visibles en sus diferentes configuraciones:

        - Escritorio: Imagen, nombre, valor y unidad.
        - Tableta: Imagen, nombre, valor y unidad.
        - Móvil: Nombre.
        - Siempre oculto: Nada.

      Únicamente se permitirá la ordenación por nombre en esta tabla.

      En la columna valor se presentará un campo de texto simple obligatorio. En

la columna de unidad se verá un combo obligatorio con las unidades disponibles en el idioma del tutorial.

- Botón de regenerar componentes:

  Será un botón por defecto de bootstrap que lanzará la función de generar la lista de componentes y cantidades del tutorial, analizando los componentes empleados en cada uno de sus pasos.

- Botón de guardar:

  Será un botón de tamaño normal de tipo principal de Bootstrap. Valida que el formulario de la primera pestaña haya sido rellenado correctamente. En caso satisfactorio, lanzará la acción de modificar tutorial en caso de estar en modo modificación o la de guardar nuevo tutorial en caso contrario.

- Botón de cancelar:

  Será un botón de tamaño normal tipo danger de Bootstrap. Cancela las modificaciones realizadas en el tutorial en la primera pestaña. Se recarga la pantalla con los mismos datos que cuando se entró por primera vez.

Diseño funcional

Las funciones que se ejecutarán en el servidor para la gestión de tutoriales serán las siguientes:

- Búsqueda de tutoriales:

  Esta función deberá ser llamada cuando se haga clic sobre el botón de búsqueda del formulario de criterios de búsqueda. Alojada en la clase controlador de Spring AdminArticulosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "buscarArticulos"
      , method = RequestMethod.POST)
   public String buscarArticulos(
        @ModelAttribute("articuloFiltro") FiltroArticulo articuloFiltro
      , Model model, Locale locale) {
   //TODO
}
```

  Esta función deberá llamar al método fncBuscarArticulos, del servicio ArticuloService, para obtener los datos de los tutoriales para el filtro de la pantalla. Una vez metida la lista de tutoriales en el modelo de Spring, se redireccionará a la pantalla de gestión de tutoriales y mostrará los resultados.

- Alta/Modificación de tutorial:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar” de la pantalla de alta/modificación de tutoriales en cualquiera de sus modos.

Alojada en la clase controlador de Spring AdminArticulosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarPaso"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String guardarPaso(
        @ModelAttribute("codArticulo") String codArticulo
      , @ModelAttribute("codigo") String codigo
      , @ModelAttribute("orden") String orden
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("descripcionComp") String descripcionComp
      , @ModelAttribute("flagImageMod") String flagImageMod
      , @ModelAttribute("imagen") String imagen
      , @ModelAttribute("componentes") String componentes
      , @ModelAttribute("valores") String valores
      , @ModelAttribute("unidades") String unidades
      , Model model
      , Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncNuevoArticulo o fncModificarArticulo, del servicio ArticuloService (dependiendo de si se está en modo alta o modificación respectivamente) para insertar o modificar el tutorial con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el tutorial se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Borrado de tutorial:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Borrar” de la fila de tutorial seleccionado. Alojada en la clase controlador de Spring AdminArticulosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "eliminarArticulo"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String eliminarArticulo(
        @ModelAttribute("codigo") String codigo
      , Model model, Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncBorrarArticulo, del servicio ArticuloService, con el código de tutorial recibido por pantalla para eliminarlo de la base de datos. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el tutorial se ha eliminado usando el componente alerts de tipo success de Bootstrap.

- Alta de paso:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar”

del wizard del formulario de alta de paso. Alojada en la clase controlador de Spring AdminArticulosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "nuevoPaso"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String nuevoPaso(
        @ModelAttribute("codArticulo") String codArticulo
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("descripcionComp") String descripcionComp
      , @ModelAttribute("imagen") String imagen
      , @ModelAttribute("componentes") String componentes
      , @ModelAttribute("valores") String valores
      , @ModelAttribute("unidades") String unidades
      , Model model
      , Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncNuevoPaso, del servicio PasoService, para insertar el paso con los parámetros recibidos por pantalla. En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el paso se ha guardado usando el componente alerts de tipo success de Bootstrap.

- Modificación de paso:

  Esta función deberá ser llamada cuando se haga clic sobre el botón “Guardar” del wizard del formulario de modificación de paso. Alojada en la clase controlador de Spring AdminArticulosController, su especificación será la siguiente:

```
@RequestMapping(
        value = "guardarPaso"
      , method = RequestMethod.POST
      , produces = "application/x-json;charset=UTF-8")
   public @ResponseBody String guardarPaso(
        @ModelAttribute("codArticulo") String codArticulo
      , @ModelAttribute("codigo") String codigo
      , @ModelAttribute("orden") String orden
      , @ModelAttribute("nombre") String nombre
      , @ModelAttribute("descripcion") String descripcion
      , @ModelAttribute("descripcionComp") String descripcionComp
      , @ModelAttribute("flagImageMod") String flagImageMod
      , @ModelAttribute("imagen") String imagen
      , @ModelAttribute("componentes") String componentes
      , @ModelAttribute("valores") String valores
      , @ModelAttribute("unidades") String unidades
      , Model model
      , Locale locale) {
   //TODO
}
```

Esta función deberá llamar al método fncModificarPaso, del servicio PasoService, para modificar el paso con los parámetros recibidos por pantalla.

En caso de que la operación en la base de datos se realice correctamente, se mostrará en la pantalla que el paso se ha guardado usando el componente alerts de tipo success de Bootstrap.

Control de excepciones

Las excepciones que ocurran durante la gestión de artículos y sus pasos se mostrarán en la sección usando el componente alerts de tipo danger de Bootstrap. Se situarán en la parte superior (encima del artículo con los criterios de búsqueda en caso de estar en la pantalla de búsqueda o de las pestañas en caso de estar en la pantalla de alta/modificación).

Plan de pruebas

- Comprobar que el rastro de migas es correcto.
- Comprobar que al entrar por URL sin estar logueado en la aplicación se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que al entrar por URL estando logueado con un usuario sin perfil de administrador o de mánager se redirige a la página de error con un mensaje de falta de permisos.
- Comprobar que, en el filtro de búsqueda, el combo de secciones se carga en función de lo que esté seleccionado en el combo de idioma.
- Comprobar que, en el filtro de búsqueda, el campo de texto para el nombre del tutorial sólo sugiere tutoriales que coincidan con lo que esté seleccionado en los combos de idioma y sección.
- Realizar una búsqueda y comprobar que los resultados coinciden con el filtro.
- Redimensionar la pantalla y comprobar que el componente de tabla cambia correctamente.
- Comprobar al pulsar en una imagen de tutorial que ésta se muestra a tamaño completo.
- Comprobar que al pulsar en una fila se despliega el menú de opciones.
- Comprobar que al pulsar el botón “Borrar”: el tutorial desaparece de la base de datos, se elimina de la tabla de resultados y se muestra el feedback correspondiente.
- Comprobar que al pulsar el botón “Nuevo” se navega a la pantalla de alta/modificación en modo alta (todos los campos están vacíos).
- Comprobar entre los datos de la primera pestaña que el combo de secciones se carga en base a lo seleccionado en el combo de idioma

- Rellenar los campos de la primera pestaña y pulsar el botón “Guardar” comprobando que valida los campos del formulario correctamente según lo especificado.
- Rellenar los campos de la primera pestaña y sin guardar navegar a la segunda pestaña y pulsar en el botón “Nuevo paso”. Comprobar que aparece una ventana avisando de que para continuar se guardarán los cambios actuales.
- Comprobar que en el formulario de alta de pasos, en el campo de descripción los componentes se muestran para el idioma y sección del paso.
- Comprobar en el formulario de alta de pasos que se valida el formulario del primer paso del wizard al intentar navegar al segundo.
- Comprobar en el formulario de alta de pasos que en el segundo paso del wizard se muestran en la tabla todos los componentes introducidos en la descripción del primer paso.
- Comprobar en el formulario de alta de pasos que al pulsar el botón “Guardar” se valida correctamente el wizard y que tras guardar se muestra el feedback de la operación y el nuevo paso en la tabla de pasos.
- Redimensionar la ventana y comprobar que la tabla de pasos muestra las columnas especificadas en las distintas resoluciones.
- Comprobar que al seleccionar un paso se despliega el menú de opciones.
- Pulsar en el botón “Editar” del paso creado y comprobar que la ventana con el wizard del paso muestra sus datos correctamente. Realizar modificaciones y comprobar que: se guardan correctamente en la base de datos, la tabla de pasos aparece actualizada y se muestra el feedback correctamente.
- Pulsar en el botón “Borrar” del paso creado y comprobar que: se borra el paso de la base de datos, desaparece de la tabla de pasos y se muestra el feedback correctamente.
- Dar de alta varios pasos para el tutorial y navegar a la pestaña de componentes.
- En la pestaña de componentes pulsar el botón “Regenerar componentes” y comprobar que la tabla de componentes se carga con el conjunto de componentes y cantidades introducidos en los distintos pasos del tutorial.
- Realizar modificaciones en las cantidades y comprobar que al pulsar el botón “Guardar” se guardan correctamente en la base de datos y se muestra el feedback.
- Pulsar el botón “Atrás” y comprobar que se muestra la pantalla de búsqueda de tutoriales.
- Pulsar en el botón “Editar” de un tutorial y comprobar que navegamos a la pantalla de alta/modificación con todos los campos cargados correctamente con

los datos que el tutorial tiene almacenados en la base de datos.

Planificación

Para la construcción de este módulo se calcula aproximadamente un 29% del total de horas dedicadas al diseño y desarrollo de la aplicación. La estimación sería la siguiente:

TOTAL (29% sobre 1000 horas aprox.)	295 horas
Consumidas en diseño	20 horas
Disponibles para implementación	275 horas

Tabla 31: Gestión de tutoriales - Planificación

4. IMPLEMENTACIÓN

4.1. Detalles de implementación

La especificación de un sistema intensivo en software tiene como ultima representación al código fuente de los componentes. Este código indica los más finos detalles del software por medio de un lenguaje preciso capaz de ser traducido automáticamente a instrucciones de la máquina.

Acompañan al código las llamadas previsiones de compilación, constituidas por todos los elementos de soporte necesarios para realizar la construcción de los componentes a partir del conjunto de códigos.

Esta sección detalla la obtención y uso del paquete de código fuente para el proyecto. Con el fin de facilitar el uso de éste para las futuras ampliaciones o correcciones del sistema.

4.1.1. Plataformas y lenguajes

El sistema operativo de la máquina servidor es la distribución de Linux Ubuntu 13.04.

El sistema de ficheros en los que se almacenará la aplicación es ext4.

El servidor de aplicaciones J2EE alojado en la máquina es un Jboss 7.1.

El servidor web empleado es un Apache v2.

El servidor de base de datos instalado es MySQL 5.5.

El lenguaje de programación a utilizar será Java en su versión de JDK 1.7. Para este lenguaje existen múltiples Frameworks que implementan el modelo de vista controlador que hemos comentado con anterioridad.

El Framework seleccionado es Spring Framework en su versión estable 3.2.0 liberada bajo licencia Apache 2.0 el 13 de diciembre de 2012.

Spring es un Framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java.

Concepto de Inversión de control:

Cuando se diseña una aplicación en Java se dispone de muchos objetos que se relacionan entre sí mediante “composición”. Para enlazar dos objetos habría que inyectarle a uno de ellos una instancia del otro. Esto lo realiza Spring automáticamente.

Por eso se llama Inversión de control, es Spring quien se encarga de estas dependencias,

instancia los objectos y los inyecta por reflexión. A grandes rasgos, se declaran en un XML los componentes de la aplicación y sus dependencias. Spring lee este XML, llamado Application Context, crea los componentes y sus relaciones entre ellos. La versión de Spring que se empleará permite anotaciones. Se puede anotar una propiedad en una clase mediante @Autowired para que Spring busque la clase correspondiente, la instancie y la inyecte, ahorrándonos bastante código XML.

Este Framework potente y moderno cada vez es más utilizado tanto en empresas privadas como en instituciones públicas y realmente ahorra mucho tiempo a la hora de codificar las aplicaciones.

Spring Framework comprende diversos módulos que proveen un rango de servicios:

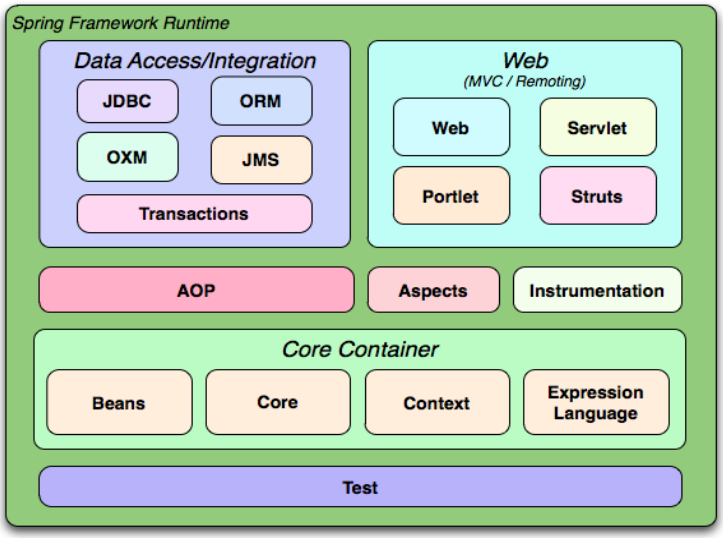


Ilustración : Módulos del Framework Spring

Los módulos de Spring que se emplearán serán los siguientes:

- Contenedor de inversión de control:

  Permite la configuración de los componentes de la aplicación y la administración del ciclo de vida de los objetos Java. Se lleva a cabo principalmente a través de la inyección de dependencias.

- Acceso a datos:

  Se trabaja con RDBMS en la plataforma java, usando Java Database Connectivity.

- Modelo vista controlador:

  Un Framework basado en HTTP y servlets que provee herramientas para la extensión y personalización de aplicaciones web y servicios web REST.

- Autenticación y Autorización:

  Procesos de seguridad configurables que soportan un rango de estándares,

protocolos, herramientas y prácticas a través del subproyecto Spring Security (formalmente Acegi Security System for Spring).

4.1.2. Obtención e instalación del paquete de fuentes

Las fuentes del proyecto se albergarán en un repositorio SVN (Subversión) instalado en la máquina servidor.

Aparte de las fuentes del proyecto también se almacenará la documentación asociada al mismo y otros archivos que guarden relación con el proyecto.

El uso del repositorio de SVN facilita:

- La gestión de versiones de los archivos del proyecto.
- La obtención de los archivos.
- La centralización de la información.

Además del repositorio de SVN se dispondrá de una carpeta en el sistema de ficheros de la máquina servidor para los despliegues de la aplicación en el servidor de aplicaciones JBoss.

4.1.3. Contenido del paquete de fuentes

Todos los archivos fuente de la aplicación estarán contenidos en un único WAR que además encapsulará las JSPs, estáticos y librerías que conforman el proyecto.

El código alfanumérico que se atribuye al proyecto es el de A83.

```
Documentación
a83War
|_ src
|    |_ a83
|         |_ beans
|         |_ comun
|         |_ controllers
|         |_ dao
|         |_ services
|_ WebContent
     |_ jsp
     |_ META-INF
     |_ staticContent
     |_ WEB-INF
          |_ lib
          |_ locales
```

4.1.4. Instrucciones de compilación

La compilación de la aplicación será copiar las fuentes de la misma a la carpeta del sistema de ficheros destinada a despliegues y después ejecutar una tarea Ant.

Apache Ant es una herramienta usada en programación para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de compilación y construcción

(build). Es, por tanto, un software para procesos de automatización de compilación desarrollado en lenguaje Java.

Esta herramienta tiene la ventaja de no depender de las órdenes del shell de cada sistema operativo, sino que se basa en archivos de configuración XML y clases Java para la realización de las distintas tareas.

La finalidad de esta tarea Ant es que se compile el código albergado en la carpeta de despliegues y a su vez se copien los archivos binarios generados a la carpeta de aplicaciones del servidor JBOSS.

Esta misma tarea deberá ser la encargada de redesplegar la aplicación en el servidor una vez que la compilación y copiado haya finalizado correctamente.

4.2. Configuración de Spring

4.2.1.1. Spring MVC WebApplicationContext

En Spring MVC, el DispatcherServlet es un Servlet que recibe las peticiones HTTP y las envía al controlador apropiado. En una aplicación SpringMVC puede haber varios DispatcherServlet para cumplir con varios propósitos (por ejemplo manejar las solicitudes de las interfaces de usuarios, solicitudes webservices, etc.). Cada DispatcherServlet tiene su propia configuración (WebApplicationContext), la cual define las características del Servlet tales como los controladores que el Servlet soporta, el manejador de mapeo, etc. También el WebApplicationContext puede incluir la configuración de la aplicación, la capa de persistencia, seguridad, servicios, etc.

4.2.1.2. DispatcherServlet y Spring Container

Como se mencionó anteriormente todas las solicitudes entrantes fluyen a través de DispatcherServlet. Como cualquier otro Servlet en una aplicación Java EE debe ser cargado en tiempo de arranque del WEB-INF/web.xml. El DispatcherServlet también es responsable de cargar un SpringApplicationContext el cual es usado para realizar el enlazado y la inyección de dependencias.

El archivo web.xml de la aplicación quedará definido de la siguiente manera:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="a83War" version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-
app_2_4.xsd">

   <context-param>
     <param-name>contextConfigLocation</param-name>
     <param-value>
         /WEB-INF/applicationContext.xml
         /WEB-INF/applicationContext-security.xml
     </param-value>
   </context-param>

   <filter>
     <filter-name>springSecurityFilterChain</filter-name>
     <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
   </filter>

   <filter-mapping>
     <filter-name>springSecurityFilterChain</filter-name>
     <url-pattern>/*</url-pattern>
   </filter-mapping>

   <filter>
     <filter-name>charsetFilter</filter-name>
     <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
     <init-param>
         <param-name>encoding</param-name>
         <param-value>UTF-8</param-value>
     </init-param>
   </filter>

   <filter-mapping>
     <filter-name>charsetFilter</filter-name>
     <url-pattern>/*</url-pattern>
   </filter-mapping>

   <listener>
     <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
   </listener>

   <servlet>
     <servlet-name>springDispatcher</servlet-name>
     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
     <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
     <servlet-name>springDispatcher</servlet-name>
     <url-pattern>/</url-pattern>
   </servlet-mapping>

   <error-page>
     <error-code>403</error-code>
     <location>/error?code=403</location>
   </error-page>

   <error-page>
     <error-code>404</error-code>
     <location>/error?code=404</location>
   </error-page>

</web-app>
```

Se registra el DispatcherServlet como un Servlet llamado springDispatcher.

Se mapea este Servlet para manejar las solicitudes entrantes que comienzan con "/".

Se usa el parámetro init ContextConfigLocation para personalizar la ubicación del archivo XML de configuración base para el SpringApplicationContext que es cargado por el DispatcherServlet en lugar de depender de la ubicación predeterminada. También será aquí donde añadimos el contexto referente a SpringSecurity.

4.2.1.3. El controlador

Ahora vamos a crear un pequeño controlador. Por ejemplo el controlador dedicado a la administración de contenidos recoge peticiones get para navegar a la administración de secciones:

```
package a83.controllers.admin;

imports ...

/**
 * Maneja peticiones de inicio de app
 */
@Controller
@RequestMapping("/admin/contenidos/")
public class AdminContenidosController implements HandlerExceptionResolver{
   protected static Logger logger = Logger.getLogger("controller");

   @Autowired
   @Qualifier(value = "seccionServiceImpl")
   private SeccionService seccionService;
   @Autowired
   @Qualifier(value = "categoriaServiceImpl")
   private CategoriaService categoriaService;
   @Autowired
   @Qualifier(value = "idiomaServiceImpl")
   private IdiomaService idiomaService;
   @Autowired
   private ReloadableResourceBundleMessageSource messageSource;

   /**
    * Navega a la página de búsqueda de secciones
    * @return the name of the JSP page
    */
   @RequestMapping(value = "secciones", method = RequestMethod.GET)
   public String gotoBusqSecciones(Model model) {
     logger.debug("AdminSecionController.gotoBusqSecciones()");

     model.addAttribute("info", "");
     model.addAttribute("infoType", "");
     model.addAttribute("migas", "Administración > Contenidos > Secciones");
     model.addAttribute("filtro", new Seccion());
     model.addAttribute("lSeccionesCombo", seccionService.fncBuscarComboSecciones());
     model.addAttribute("lIdiomas", idiomaService.fncBuscarIdiomas());
     model.addAttribute("lSecciones", new ArrayList<Map<String,Object>>());
     model.addAttribute("lSeccionesJson", new JSONArray());

     // This will resolve to /adminJSP/seccionJSP/busqueda.jsp
     return "seccionesAdminJSP";
   }
}
```

Ahora vamos a describir ciertos aspectos de esta clase:

La clase utiliza la anotación @Controller indicando que es un controlador de Spring MVC capaz de manejar solicitudes web. Esta clase será automáticamente detectada por el contenedor de Spring.

En el método gotoBusqSecciones() se utilizó la anotación @RequestMapping para especificar que este método debe manejar las solicitudes web con el path "secciones". Si nos fijamos, el propio controlador también posee la misma propiedad que define a modo de prefijo del resto la URL declarada "/admin/contenidos/". Con lo cual el método gotoBusqSecciones() de este controlador recibirá del dispatcher de spring las peticiones get a "/admin/contenidos/secciones".

El método simplemente retorna el String "seccionesAdminJSP" indicando la página definida en el archivo de apache tiles que se debe manejar en la respuesta.

```
<definition name="seccionesAdminJSP" extends="plantilla">
  <put-attribute name="conf" value="/jsp/adminJSP/seccionJSP/include/busquedaConf.jsp" />
  <put-attribute name="body" value="/jsp/adminJSP/seccionJSP/busqueda.jsp" />
</definition>
```

4.2.1.4. Spring Application Context

Finalmente, necesitamos crear un archivo básico de definición Spring ApplicationContext.

WEB-INF/applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
 [...]
>
  <!-- Handlers for use annotations -->
  <bean
class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping" /
>
  <bean
class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter" />

  <!-- Activates various annotations to be detected in bean classes -->
  <mvc:annotation-driven />
  <context:property-placeholder location="/WEB-INF/a83.properties" />

  <bean id="properties"
class="org.springframework.beans.factory.config.PropertiesFactoryBean">
    <property name="locations">
      <list><value>/WEB-INF/a83.properties</value></list>
    </property>
  </bean>
  <import resource="jdbc-context.xml" />
  <!-- Scans the classpath for annotated components that will be auto-registered
    as Spring beans. For example @Controller and @Service. Make sure to set the
    correct base-package -->
  <context:component-scan base-package="a83" />
  <!-- i18n -->
  <bean id="localeResolver"
class="org.springframework.web.servlet.i18n.SessionLocaleResolver">
    <property name="defaultLocale" value="es" />
  </bean>
  <!-- interceptor para cambiar la locale -->
  <mvc:interceptors>
    <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor"
          p:paramName="lang" />
  </mvc:interceptors>
  <!-- Register the comun.properties -->
  <bean id="messageSource"

  class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
    <property name="basenames" value="/WEB-INF/locales/comun" />
    <property name="defaultEncoding" value="UTF-8"/>
  </bean>
  <!-- Maneja las peticiones GET a /staticContent/** para servir el contenido estático -->
  <mvc:resources mapping="/staticContent/**" location="/staticContent/" />
  <!-- Configure the multipart resolver -->
  <bean id="multipartResolver"
        class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
    <!-- one of the properties available; the maximum file size in bytes -->
    <property name="maxUploadSize" value="1073741824" />
  </bean>

  <bean id="webexpressionHandler"
class="org.springframework.security.web.access.expression.DefaultWebSecurityExpressionHa
ndler"/>
</beans>
```

Examinemos el contenido de este archivo.

La declaración <context:component-scan> le dice a Spring que escanee el código para encontrar beans inyectables bajo el paquete y todos sus subpaquetes especificados. De este modo cualquier código anotado con @Controller es automáticamente descubierto.

La declaración <mvc:annotation-driven> habilita el soporte de anotaciones para Spring

MVC como lo es por ejemplo la anotación @Controller.

La declaración <context:property-placeholder> expone el archivo de propiedades de la aplicación alojado en WEB-INF/a83.properties para su acceso mediante objetos EL. En este archivo se encuentran aquellos parámetros necesarios para la implantación de la aplicación, rutas, cadenas de conexión, drivers, etc.

La parte <import resource="jdbc-context.xml" /> sirve para incluir la parte abstraída del contexto de spring referente a Spring JDBC encargada de la persistencia de datos. Este archivo configura el transactionManager encargado de manejar las transacciones con la base de datos. Ésta es su definición:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:context="http://www.springframework.org/schema/context"
  xmlns:tx="http://www.springframework.org/schema/tx"
  xmlns:p="http://www.springframework.org/schema/p"
  xsi:schemaLocation="
            http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
            http://www.springframework.org/schema/tx
            http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
            http://www.springframework.org/schema/context
            http://www.springframework.org/schema/context/spring-context-3.0.xsd">

  <!-- Enable annotation style of managing transactions -->
  <tx:annotation-driven transaction-manager="transactionManager" />

  <!-- Declare a datasource that has pooling capabilities -->
  <bean id="dataSource"
        class="org.apache.commons.dbcp.BasicDataSource"
        destroy-method="close"
        p:driverClassName="${app.jdbc.driverClassName}"
        p:url="${app.jdbc.url}"
        p:username="${app.jdbc.username}"
        p:password="${app.jdbc.password}" />

  <!-- Declare a transaction manager -->
  <bean id="transactionManager"
        class="org.springframework.jdbc.datasource.DataSourceTransactionManager"
        p:dataSource-ref="dataSource" />
</beans>
```

En ella se puede ver como se define el DataSource empleado con las propiedades de la aplicación expuestas como objetos de EL en el SpringApplicationContext como se ha explicado anteriormente.

4.3. Optimización

Las páginas lentas son un serio problema. La usabilidad se reduce a cero pese a todos los esfuerzos realizados en el diseño y estructura si la velocidad de carga de la página web es lenta. Los contenidos no se leen porque el usuario abandona la página antes de profundizar en ella.

En 2010 Google agregó la velocidad de carga de un sitio web como uno de los

elementos de su algoritmo de búsqueda, sumándolo a los alrededor de 200 factores que se utilizan actualmente. Eso significa que hay que preocuparse de la velocidad de carga de la aplicación web si se desea conseguir un buen SEO (Posicionamiento en buscadores) para ella. Google considera que los usuarios prefieren sitios web que se demoren poco en cargar, así que cuanto más rápido pueda desplegar el contenido, mejor.

4.3.1. Factores que determinan el tiempo de carga

Vamos a ver algunos de los factores que más influyen en el tiempo de carga de una web:

- **Hosting:**

  Sin duda es el factor principal. Es importante dónde está ubicado el servidor, la latencia del mismo, el número de saltos (tracert) hasta llegar al mismo, el tiempo de respuesta, la velocidad del disco duro que tiene instalado, los recursos del servidor (RAM, Procesadores, etc), tipo de tarjeta de red, Sistema Operativo, ancho de banda asignado, etc. Un hosting de calidad es necesario, a ser posible en el mismo país hacia el que va dirigida la web, con unas características suficientes y a ser posible dedicado (muy importante este detalle).

  En estos momentos al no contar con financiación para el proyecto se cuenta con un ordenador personal muy limitado y que no está exclusivamente dedicado a las tareas de hosting.

- **Red:**

  Es importante la red donde está conectada nuestro servidor, al igual que la red a la que se conectan los usuarios. Que el centro de datos tenga contratada una red de calidad es muy recomendable, tanto por fiabilidad como por velocidad. A veces se redireccionan las peticiones a otros nodos por averías en la red o mantenimiento y se pasa de tener un tracert de 8 saltos a tener uno de 13 saltos, cuanto menos saltos se necesiten en estas averías, mejores tiempos de respuesta tendrán los usuarios. El operador de ADSL (o similares) con el que trabajen los usuarios también influirá en la velocidad a la que ellos cargarán nuestra página web. Este factor escapa totalmente a nuestro control.

  Al igual que en el apartado de hosting, no se cuenta con un ancho de banda adecuado por falta de financiación. Actualmente el host está conectado con un ancho de banda reducido gracias un contrato de ISP a particulares con Movistar. El servicio contratado da unas velocidades constantes de 10Mb de bajada y 820 Kbps de subida.

- **Programación:**

  Otro factor muy importante. En igualdad de condiciones, una web bien programada y optimizada (sobre todo en temas de diseño y consultas a bases de datos, correcto uso de la cache, etc.) influye enormemente en la velocidad de carga de la misma. Hay que cuidar cada vez más estos detalles y así se ha

intentado a lo largo del desarrollo del proyecto.

4.3.2. Decisiones tomadas para optimizar la aplicación

A continuación se detallan las acciones que se han tomado respecto de la optimización de la aplicación en el desarrollo de la misma.

4.3.2.1. Gestión de la caché

Para la gestión de la caché se especifican los siguientes "interceptors" en el archivo de configuración del contexto de Spring "applicationContext.xml":

```
<mvc:interceptor>
   <mvc:mapping path="/**/*.js" />
   <bean id="webContentInterceptorJS"
         class="org.springframework.web.servlet.mvc.WebContentInterceptor">
      <property name="cacheSeconds" value="604800" />
      <property name="useExpiresHeader" value="true" />
      <property name="useCacheControlHeader" value="true" />
      <property name="useCacheControlNoStore" value="true" />
   </bean>
</mvc:interceptor>
```

Este "interceptor" especifica un tiempo de cacheo de una semana (604.800 sg) para los archivos de javascript que se sirvan (*.js).

```
<mvc:interceptor>
   <mvc:mapping path="/**/*.css" />
   <bean id="webContentInterceptorCSS"
         class="org.springframework.web.servlet.mvc.WebContentInterceptor">
      <property name="cacheSeconds" value="604800" />
      <property name="useExpiresHeader" value="true" />
      <property name="useCacheControlHeader" value="true" />
      <property name="useCacheControlNoStore" value="true" />
   </bean>
</mvc:interceptor>
```

Este "interceptor" especifica un tiempo de cacheo de una semana (604.800 sg) para los archivos de estilos que se sirvan (*.css).

```
<mvc:interceptor>
   <mvc:mapping path="/**/*.jpg" />
   <mvc:mapping path="/**/*.gif" />
   <mvc:mapping path="/**/**.png" />
   <mvc:mapping path="/**/**.svg" />
   <bean id="webContentInterceptorImages"
         class="org.springframework.web.servlet.mvc.WebContentInterceptor">
      <property name="cacheSeconds" value="604800" />
      <property name="useExpiresHeader" value="true" />
      <property name="useCacheControlHeader" value="true" />
      <property name="useCacheControlNoStore" value="true" />
   </bean>
</mvc:interceptor>
```

Este “interceptor” especifica un tiempo de cacheo de una semana (604.800 sg) para los archivos de imagen que se sirvan (*.jpg, *.gif, *.png y *.svg) .

```
<mvc:interceptor>
   <mvc:mapping path="/**/*.ttf" />
   <mvc:mapping path="/**/*.woff" />
   <bean id="webContentInterceptorFonts"
         class="org.springframework.web.servlet.mvc.WebContentInterceptor">
      <property name="cacheSeconds" value="604800" />
      <property name="useExpiresHeader" value="true" />
      <property name="useCacheControlHeader" value="true" />
      <property name="useCacheControlNoStore" value="true" />
   </bean>
</mvc:interceptor>
```

Este “interceptor” especifica un tiempo de cacheo de una semana (604.800 sg) para los archivos de fuentes de texto que se sirvan (*.ttf y *.woff).

4.3.2.2. Compresión de archivos CSS y JS

Dada la gran cantidad de componentes de terceros empleados en la aplicación la cantidad de archivos de JavaScript y CSS es elevada (y relativamente pesada). Una buena práctica para la optimización de páginas web es la unificación, minificación y compresión de estos archivos.

Para la consecución de esta tarea se optó por el proyecto WRO4J (Web Resource Optimizer for Java). Es un proyecto java gratuito y de código abierto que podemos encontrar en Google Code (http://code.google.com/p/wro4j/). El proyecto fue desarrollado con el ánimo de agrupar y simplificar la configuración y uso de las distintas tecnologías y algoritmos libres disponibles.

Permite el uso de herramientas como JsHint, CssLint, JsMin, Google Closure compressor, YUI Compressor, UglifyJs, Dojo Shrinksafe, Css Variables Support, JSON Compression, Less, Sass, CoffeeScript y muchas más. Además se integra muy bien con Spring.

Para su configuración se añade el filtro al archivo “web.xml”:

```
<filter>
   <filter-name>WebResourceOptimizer</filter-name>
   <filter-class>
      org.springframework.web.filter.DelegatingFilterProxy
   </filter-class>
   <init-param>
      <param-name>targetBeanName</param-name>
      <param-value>wroFilter</param-value>
   </init-param>
   <init-param>
      <param-name>targetFilterLifecycle</param-name>
      <param-value>true</param-value>
   </init-param>
</filter>

<filter-mapping>
   <filter-name>WebResourceOptimizer</filter-name>
   <url-pattern>/staticContent/wro/*</url-pattern>
</filter-mapping>
```

De esta forma los estáticos se obtendrán a partir de la ruta "/staticContent/wro/*" y serán manejados por la herramienta WRO4J.

Por otra parte en el contexto de spring "applicationContext.xml" declaramos el filtro y especificamos que su configuración se realizará a través de un archivo de propiedades "wro.properties":

```
<bean id="wroFilter" class="ro.isdc.wro.http.ConfigurableWroFilter">
   <property name="properties" ref="wroProperties" />
</bean>

<bean id="wroProperties"
      class="org.springframework.beans.factory.config.PropertiesFactoryBean">
   <property name="location">
      <value>/WEB-INF/wro.properties</value>
   </property>
</bean>
```

En este archivo de propiedades del WRO4J la configuración empleada será la siguiente:

```
debug=false
minimizeEnabled=true
gzipEnabled=true
jmxEnabled=true
mbeanName=wro
cacheUpdatePeriod=604800
modelUpdatePeriod=604800
disableCache=false
encoding=UTF-8
preProcessors=yuiCssMin,googleClosureSimple
```

En este archivo entre otros parámetros, se especifican que los archivos a generar se minimizarán y comprimirán mediante las librerías yuiCssMin y googleClosureSimple. Además especificamos que el período de cacheo será de una semana (604.800 sg).

Habiendo llegado a este punto, únicamente hace falta cumplimentar el archivo "wro.xml" que especifica el guión de los archivos JavaScript y CSS que se generarán

mediante WRO4J. Para el proyecto únicamente se crearán tres archivos de estáticos (uno para JavaScript, otro para CSS y uno separado para las validaciones JavaScript) de la siguiente forma:

```
<?xml version="1.0" encoding="UTF-8"?>
<groups  xmlns="http://www.isdc.ro/wro"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.isdc.ro/wro wro.xsd">

   <group name="validate">
      <js minimize="true">/staticContent/js/jquery.validate.js</js>
   </group>

   <group name="grupo">
      <css minimize="true">/staticContent/css/bootstrap.css</css>
      [...]

      <js minimize="true">/staticContent/js/bootstrap.js</js>
      [...]
   </group>
</groups>
```

De esta forma los estáticos producidos (ya optimizados) se encontrarán en las siguientes rutas:

- /staticContent/wro/validate.js
- /staticContent/wro/grupo.js
- /staticContent/wro/grupo.css

4.3.3. Aplicaciones de Testeo

Una vez conocidos los factores que determinan el tiempo de carga de la página web, es hora de ver las herramientas para poder hacer dicho proceso.

Google PageSpeed Insights:

Construida por Google (https://developers.google.com/speed/pagespeed/insights/) es de las más exigentes que podemos encontrar en la Web y emplea los mismos algoritmos que utiliza su famoso buscador para el posicionamiento SEO.

Pingdom Tools:

Pingdom Tools es, sin duda, la herramienta online mas completa para medir y mejorar la velocidad de carga de una página web. Su funcionamiento consiste en testear la página web por completo rastreando todo el contenido a través del código HTML (es capaz de detectar imágenes, CSS, Javascripts, RSS, Flash y frames/iframes). De hecho, imita la manera en la cual se cargaría la web normalmente. El funcionamiento es realmente sencillo, ingresamos la URL y le damos clic en “Test now!”. Además de forma gratuita,

permite medir 3 bloques importantes de la página web:

- Test total de la página web: Aparte de mostrar el tiempo total de carga de la página web, muestra individualmente cada uno de los elementos de la página (en forma de gráfico de barras), desde que hacen contacto con el servidor hasta que se descargan.
- Salud DNS: Comprobación rápida de la conexión con el servidor, algo muy útil cuando se tienen problemas de carga.
- Ping y Traceroute: Son comandos muy útiles que permiten acceder a información básica del equipo de hosting para poder, por ejemplo, comunicarse vía remota o con otros equipos de la red.

GTmetrix:

Es la mejor herramienta de todas ya que nos muestra de manera porcentual el impacto de cada error de la página web y su correspondiente solución de forma detallada y fácil de corregir.

5. SEGUIMIENTO Y CONTROL DEL PROYECTO

En este apartado se tratará de obtener una visión objetiva del estado actual del proyecto y determinar las posibles desviaciones a fin de realizar las correcciones necesarias.

5.1. Seguimiento del alcance

Durante la consecución del proyecto han sucedido varios acontecimientos que han variado el alcance definido en el análisis de requisitos restando servicios secundarios a la aplicación.

Los cambios realizados pueden observarse en el apartado de "Cambios acontecidos".

Las fechas de finalización planificadas se han mantenido como estaban planificadas.

5.2. Seguimiento temporal y de costes

En este apartado se van a analizar las desviaciones de coste y tiempo que se han producido respecto a la planificación del proyecto.

5.2.1. Retrasos acontecidos

Los retrasos en el desarrollo del proyecto se han producido principalmente por encontrar dificultades al emplear diferentes componentes en la interfaz y conseguir que estos funcionen entremezclándose de manera responsiva en diferentes dispositivos y resoluciones.

Cabe destacar que he estado compaginando el desarrollo de este proyecto con la vida laboral diaria, la cual en ocasiones ha requerido sobre-esfuerzos aleatorios que restaban horas de dedicación a este proyecto.

5.2.2. Horas planificadas versus horas reales

No ha habido desvío horario puesto que el problema de tiempos se encontraba en la tarea "(F5.5) Diseño e implementación de la aplicación" que constaba de una asignación de 1.000 horas para la consecución de todos los requisitos especificados.

La forma de mantener invariables las horas planificadas y por tanto la fecha de finalización ha sido la supresión de ciertos requisitos de baja prioridad.

De esta forma las horas destinadas al proyecto se han mantenido en las **1,535 horas** iniciales.

5.2.3. Costes planificados versus costes reales

Dado que no ha habido modificaciones en las horas planificadas, los costes del proyecto permanecen sin alteraciones en **32.870,99 €**.

5.3. Gestión de calidad

5.3.1. Introducción

Para realizar un correcto control de calidad es necesario el uso de una serie de herramientas que den la seguridad de que se está llevando a cabo un trabajo que cumpla con los niveles de calidad planificados. De esta forma será factible encauzar las posibles desviaciones que puedan acontecer.

La calidad mínima en este proyecto ha sido satisfecha e incluso se han alcanzando niveles de calidad superiores, como así se planificó y quedó definido en el alcance.

La prevención temporal, del 14% del tiempo total, ha sido utilizada puesto que no se posee experiencia alguna en muchas de las tecnologías seleccionadas. Es cierto que las horas disponibles para la consecución del proyecto no fueron suficientes para las pretensiones que se tenían en el mismo, y se tuvo que reducir el alcance. De todas formas la calidad de las funcionalidades sí implementadas no se ha visto reducida.

5.3.2. Aseguramiento y control de calidad

Tal y como se planificó, para verificar el cumplimento de calidad y de la consistencia de las tareas realizadas en el proyecto se han realizado inspecciones generales y específicas de cada aplicación en los correspondientes seguimientos tras la finalización de cada una de ellas.

Cabe destacar que en las ejecuciones del plan de pruebas la mayor parte de los errores encontrados han estado en la interfaz. Muchos de estos errores han sido en base a problemas derivados de la conjunción de los distintos componentes de jQuery empleados en el proyecto y de la adaptación responsiva de las distintas pantallas.

5.4. Control de riesgos

Durante la ejecución del proyecto se han sufrido una serie de riesgos incluidos en la planificación. Éstos, junto con los cambios que se han llevado a cabo para paliarlos han derivado en recortes en el alcance del proyecto.

En las siguientes tablas se definen los riesgos acontecidos así como la solución impuesta para solucionarlos y poder continuar con el proyecto de forma eficaz.

Riesgo: **Retardo en el aprendizaje de nuevas tecnologías**			
Probabilidad: 30%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Puede influir en la consecución del proyecto			
Plan de contingencias:	Para el desarrollo del proyecto es necesario tener un conocimiento pleno de las tecnologías empleadas tanto para el soporte como para la implementación del mismo. Así las futuras tareas serán realizadas de manera correcta y se evitará una desviación aún más importante.		
¿Ha surgido?:	**Sí**		
Suceso acontecido:	Han surgido nuevos componentes a incluir que no estaban contemplados en el estudio previo y que requerían cierto aprendizaje de uso.		
Solución efectuada:	En este caso no ha sido necesaria ninguna solución, puesto que el tiempo empleado en el aprendizaje de los nuevos componentes era sensiblemente menor que el que se hubiese tenido que emplear en un desarrollo propio.		

Tabla 32: Control de riesgos (I)

Riesgo: **Pérdida de conexión con el servidor o caída del mismo**			
Probabilidad: 40%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Imposibilidad de conexión con la aplicación de forma remota			
Plan de contingencias:	Se contará con un equipo portátil con todos los servicios necesarios instalados y configurados de tal forma que la aplicación sea funcional en el mismo entorno de desarrollo en caso de ser necesario.		
¿Ha surgido?:	**Sí**		
Suceso acontecido:	A menudo ha habido problemas de conexión con el servidor del entorno de producción. Algunas veces causadas por el proveedor del dominio web y otras por problemas en el mismo servidor.		
Solución efectuada:	Se han tenido que corregir estos problemas cuando se ha necesitado pasar el plan de pruebas desde dispositivos que necesitaban el uso del entorno de producción. En el resto de casos se han llevado a cabo desde el entorno de desarrollo.		

Tabla 33: Control de riesgos (II)

Riesgo: **Elección equivocada de las herramientas de trabajo**			
Probabilidad: 10%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Es posible que con alguna de las tecnologías elegidas no sea posible cumplir los objetivos planteados o dar a la aplicación la funcionalidad deseada.			
Plan de contingencias:	Realizar un estudio profundo sobre las diferentes tecnologías y tener planteadas tecnologías alternativas para comenzar a utilizarlas lo antes posible.		
¿Ha surgido?:	**No**		
Motivo:	Desde el inicio del proyecto se tuvo claro que herramientas utilizar para realizar el desarrollo del mismo, es por esto que no se ha producido tal riesgo.		

Tabla 34: Control de riesgos (III)

Riesgo: **Estancamiento en la codificación**			
Probabilidad: 50%	Controlado: SI	Previsible: NO	Origen: Interno
Impacto: Podría llegar a darse algún problema de difícil solución que paralizaría momentáneamente el avance del proyecto.			
Plan de contingencias:	Se ha dejado de margen el 14% de las horas planificadas para posibles situaciones como ésta.		
¿Ha surgido?:	Sí		
Suceso acontecido:	La falta de conocimientos en el Framework de Spring causó fallos originados en la arquitectura de la aplicación una vez había comenzado el desarrollo.		
Solución efectuada:	Hubo que parar el desarrollo y volver a retomar la documentación de Spring para averiguar cual era el problema y dejar la arquitectura correctamente. Se estima la desviación en unas 12 horas.		
Suceso acontecido:	Los componentes de presentación elegidos tienen problemas de compatibilidad entre ellos.		
Solución efectuada:	Dado que todos los componentes seleccionados eran de código abierto se han realizado modificaciones en los mismos de tal forma que puedan relacionarse correctamente entre todos ellos sin problemas. Se calcula una desviación de unas 150 horas. En primera instancia puede parecer un desfase alto, pero se calcula que el tiempo que se invertiría en buscar componentes de repuesto o un desarrollo propio sería mucho más alto.		

Tabla 35: Control de riesgos (IV)

Riesgo: **Baja del desarrollador de la aplicación**			
Probabilidad: 8%	Controlado: NO	Previsible: NO	Origen: Interno
Impacto: Podría suceder que por enfermedad o accidente, el desarrollador del proyecto quede incapacitado durante un período de tiempo provocando la paralización del proyecto.			
Plan de contingencias:	No existe.		
¿Ha surgido?:	**No**		
Motivo:	No ha habido problemas de salud.		

Tabla 36: Control de riesgos (V)

Riesgo: **Análisis erróneo o planificación muy optimista**			
Probabilidad: 60%	Controlado: SÍ	Previsible: SÍ	Origen: Interno
Impacto: Una mala planificación podría acarrear retrasos en el desarrollo del proyecto.			
Plan de contingencias:	Modificar la planificación alargando la duración del proyecto o disminuyendo el alcance del mismo.		
¿Ha surgido?:	**Sí**		
Suceso acontecido:	El alcance inicial del proyecto era demasiado grande. Tal y como se planificó era un suceso previsible y la probabilidad de que sucediese, alta. La inexperiencia en el uso de las tecnologías escogidas podía resultar en desviaciones en las horas por errores y tiempos de desarrollo elevados.		
Solución efectuada:	Se ha disminuido el alcance del proyecto de forma que las funcionalidades con menos peso en la aplicación han sido suprimidas. Estos datos pueden consultarse en el apartado “Cambios acontecidos”.		

Tabla 37: Control de riesgos (VI)

Riesgo: **Pérdida de los datos del proyecto por fallo del hardware o software maligno**			
Probabilidad: 10%	Controlado: NO	Previsible: NO	Origen: Interno
Impacto: Debido a un virus o a un fallo en los componentes del ordenador en el que se desarrolla el proyecto podría ocasionar una pérdida total o parcial de los datos del mismo.			
Plan de contingencias:	Tener copias de seguridad del proyecto en distintos dispositivos e instalar un antivirus y cortafuegos en el sistema.		
¿Ha surgido?:	**No**		
Motivo:	Se han realizado "Backups" de código fuente, base de datos y documentación de proyecto.		

Tabla 38: Control de riesgos (VII)

5.5. Cambios acontecidos

Como se ha comentado anteriormente no todos los requisitos de la aplicación han podido implementarse por una mala estimación en los tiempos de diseño e implementación de la aplicación. La construcción de la aplicación se estimó en **1.000 horas** para que el proyecto pudiese llevarse a cabo sin mover la fecha de entrega. En su momento parecía una suma holgada para su consecución, pero se hizo oportuna una disminución del alcance del proyecto. En el transcurso del desarrollo de la tarea, tras ver realmente los tiempos destinados al desarrollo, surgieron problemas originados principalmente con los componentes de interfaz. Todos ellos tenían que funcionar correctamente entre sí en un entorno responsivo, causando un aumento de las horas estimadas.

La decisión de cambiar el alcance del proyecto estaba ya contemplada en el análisis de riesgos y así se puede ver la tabla "Estancamiento en la codificación" del punto 4.5 (control de riesgos) de este apartado.

Es por ello que se optó por no atender a los siguientes requisitos del análisis:

- **Búscar tutorial**

  En lugar de lo especificado en el requisito se implementa una sencilla búsqueda por sección y nombre de tutorial para permitir a los usuarios encontrar los tutoriales.

- **No funcional - Uso de cookies**

  Requisito con prioridad baja que ha sido descartado por completo.

Además de esto se ha suprimido la realización del manual de usuario estimada en 20 horas. Gracias a la omisión total o parcial de estos requisitos ha sido posible mantener las horas destinadas a diseño e implementación en 1000 horas y terminar el proyecto en fechas, por lo que se entienden como cambios más que justificados.

5.6. Control de adquisiciones

Durante la realización del proyecto se han llevado las siguientes adquisiciones.

Se han empleado componentes de presentación realizados por terceros con licencias de MIT o BSD que permiten el uso comercial de las mismas, realizar modificaciones, etc. como se ha comentado anteriormente.

A continuación se describen los componentes de presentación adquiridos.

5.6.1. TouchSwipe

Dado que se ha planteado una interfaz responsiva capaz de ser usable en los dispositivos móviles actuales, no hay que olvidar que hoy día la mayoría de estos gadgets poseen pantallas táctiles. Es por esta razón por la se hace necesario el uso de esta librería de jQuery, la cual nos da la capacidad de poder responder a gestos táctiles sobre la interfaz.

Sus características principales son:

- Detecta deslizamientos en 4 direcciones, "arriba", "abajo", "izquierda" y "derecha"
- Detecta pellizcos "dentro" y "fuera"
- Soporta eventos táctiles hasta para dos dedos.
- Soporta eventos de clic tanto en el objeto touchSwipe y sus descendientes.
- Se puede definir el umbral de deslizamiento que diferencia entre deslizamiento y toque.
- Un único deslizamiento activa los eventos de "inicio", "movimiento", "fin" y "cancelar"
- El evento final puede ser disparado tanto en versión táctil (al acabar el gesto) o tan pronto como se alcance el umbral definido.
- Permite deslizar y hacer scroll de página.

Web de descarga: https://github.com/mattbryson/TouchSwipe-Jquery-Plugin

Autor: Matt Bryson

5.6.2. BlockUI

BlockUI es un plugin de jQuery que permite bloquear la página presentada, por completo o de forma parcial. Este plugin es muy sencillo y ampliamente personalizable.

Web de descarga: https://github.com/malsup/blockui/

Autor: Mike Alsup

5.6.3. SlitSlider

Este plugin de jQuery nos ofrece la posibilidad de crear sliders. Los sliders son un carrusel de capas que representan bloques de información de forma muy gráfica. La peculiaridad de este carrusel es que al pasar de una diapositiva a otra se corta la primera en dos rebanadas con un resultado muy cuidado y fluido.

Es cierto que entre los plugins contenidos en la librería de Bootstrap se encuentra uno dedicado a la creación de carruseles. Pero el elegido SlitSlider aparte de ser responsivo como el de bootstrap es más potente y vistoso.

Web de descarga: http://tympanus.net/codrops/2012/06/05/fullscreen-slit-slider-with-jquery-and-css3/

Autor: Codrops

5.6.4. Fit Text

Este plugin es muy útil en diseños sensibles puesto que se encarga de que el texto contenido en sus objetos sean flexibles. De esta forma podemos realizar titulares o cabeceras escalables dependiendo del tamaño de pantalla disponible.

Web de descarga: https://github.com/davatron5000/FitText.js

Autor: Paravel

5.6.5. FooTable

Este plugin es muy útil para la presentación de tablas en un diseño sensible o responsivo. Las tablas siempre son un problema en este tipo de diseños puesto que por mucho que el ancho de la tabla se ajuste al ancho de la pantalla el número de columnas y sus contenidos siempre son un bache en el camino.

Este componente permite ocultar columnas en función de la resolución de la pantalla, presentando las columnas ocultas a través de una capa que se despliega de cada fila de tabla.

Web de descarga: http://themergency.com/footable/

Autor: themergency.com

5.6.6. Bootstrap Dialog

Bootstrap de por sí ofrece el componente de ventanas modales, pero su uso se hace costoso en la parte declarativa de las mismas. Con este plugin se añade una capa de jQuery por encima a la de Bootstrap que permite generar este tipo de ventanas modales mediante el uso de funciones JavaScript.

Web de descarga: http://nakupanda.github.io/bootstrap3-dialog/

Autor: nakupanda

5.6.7. Bootstrap MaxLength

Este plugin se integrá con Bootstrap para mostrar al usuario el tamaño máximo que admite el campo de texto sobre el cual está insertando caracteres. El indicador se muestra cuando el campo de texto consigue el foco y se oculta cuando lo pierde.

Web de descarga: http://mimo84.github.io/bootstrap-maxlength/

Autor: Maurizio - mimo84

5.6.8. jQuery Mentions

Es un plugin de jQuery que permite realizar menciones en campos de texto como las que admiten las redes sociales actuales. Permite la realización de peticiones Ajax para conseguir los datos de las menciones y el uso de imágenes en las mismas.

Web de descarga: http://podio.github.io/jquery-mentions-input/

Autor: Kenneth Auchenberg

5.6.9. jQuery Share In1

Es un plugin de jQuery que permite la conexión con los distintos servicios REST de publicado en las redes sociales más comunes de una forma sencilla.

Web de descarga: http://plugins.in1.com/share

Autor: Carol Skelly

5.6.10. Bootstrap wizard

Este plugin ofrece una capa superior a la del componente de pestañas de Bootstrap, convirtiendo las mismas en un wizard de pasos secuenciales.

Web de descarga: https://github.com/VinceG/twitter-bootstrap-wizard

Autor: Vincent Gabriel

5.7. Control de optimización

A continuación se muestran los resultados de las pruebas de testeo tras las modificaciones acontecidas para optimizar los tiempos de carga de la aplicación. Así mismo se comentan las advertencias o recomendaciones que nos plantean las herramientas de testeo seleccionadas.

5.7.1. Google PageSpeed Insights

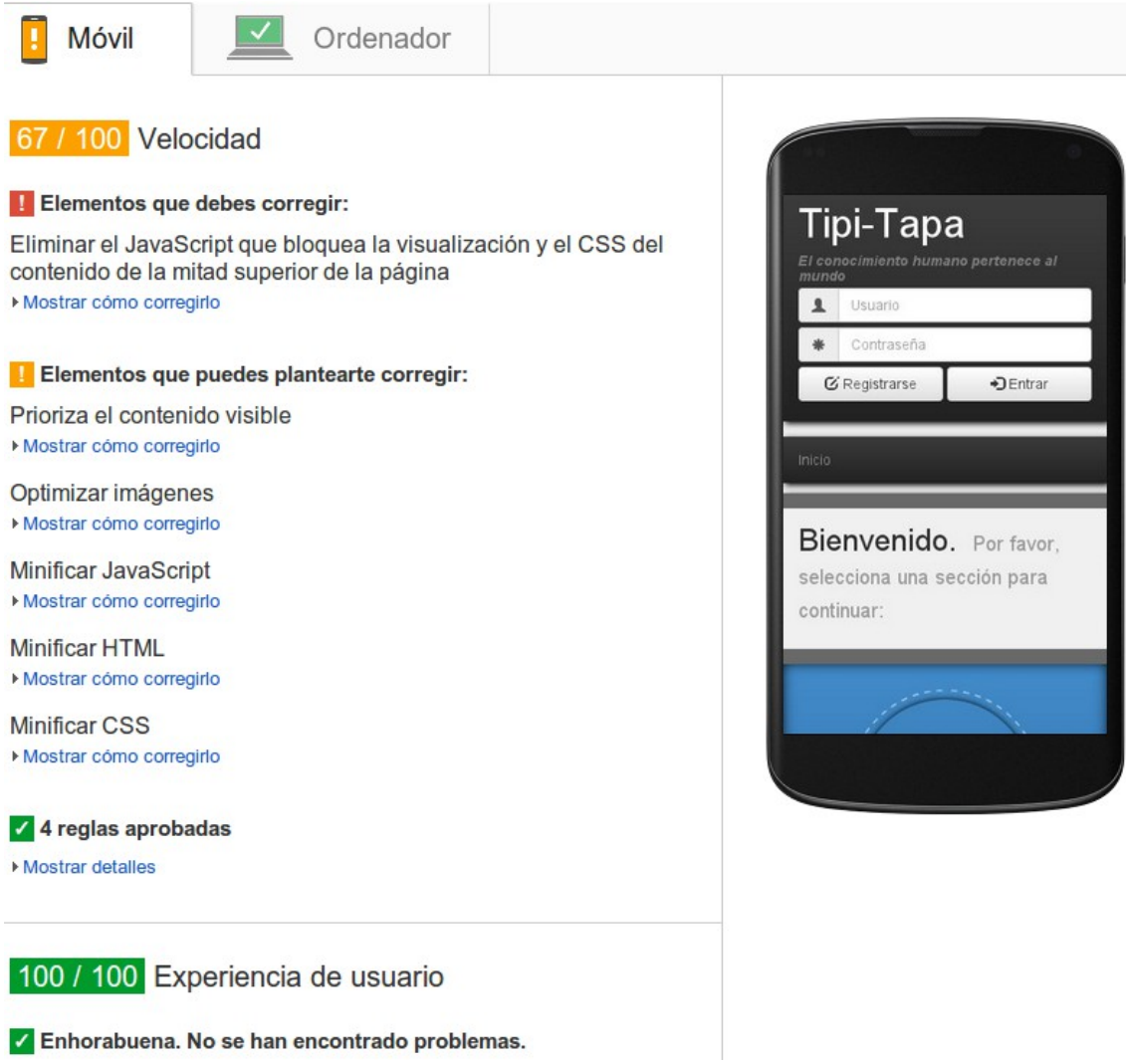


Ilustración 56: Control de optimización - Google PageSpeed Insights (móvil)

Las puntuaciones de Google no son tan buenas porque tiene muy en cuenta el tiempo de respuesta del servidor, problema que como ya se ha explicado anteriormente no es

mejorable sin financiación.

- Optimizar imágenes:

  Formatear y comprimir correctamente las imágenes puede ahorrar una gran cantidad de bytes de datos.

  Optimizar estas imágenes para reducir su tamaño en 14,1 KB (reducción del 8%).

  Se considera un porcentaje demasiado bajo como para invertir tiempo en su corrección.

- Minificar JavaScript:

  Compactar el código JavaScript puede ahorrar una gran cantidad de bytes de datos y acelerar los tiempos de descarga, análisis y ejecución.

  Reducir JavaScript de estos recursos para reducir su tamaño en 3,6 KB (reducción del 3%).

  Se considera un porcentaje demasiado bajo como para invertir tiempo en su corrección.

- Minificar HTML:

  Compactar el código HTML, incluido cualquier código CSS y JavaScript insertado que contenga, puede ahorrar una gran cantidad de bytes de datos y acelerar los tiempos de descarga y análisis.

  Reducir HTML de estos recursos para reducir su tamaño en 1,4 KB (reducción del 21%).

  Este porcentaje se considera alto. La programación de las JSPs se ha realizado de forma visualmente correcta (con sangrías y espacios que hacen fácilmente comprensible la codificación) pero funcionalmente no aporta nada y además es una carga en el peso de las páginas.

  No se ha tenido en cuenta ninguna tecnología de compresión de HTML a la hora de compilar las JSPs en el servidor. El tiempo que se estima necesario para su realización es tan elevado que no se considera viable para la entrega del proyecto. Eso sí, sería interesante solucionarlo en el futuro.

- Minificar CSS:

  Compactar el código CSS puede ahorrar una gran cantidad de bytes de datos y acelerar los tiempos de descarga y análisis.

  Reducir CSS de estos recursos para reducir su tamaño en 1,1 KB (reducción del 4%).

  Se considera un porcentaje demasiado bajo como para invertir tiempo en su corrección.

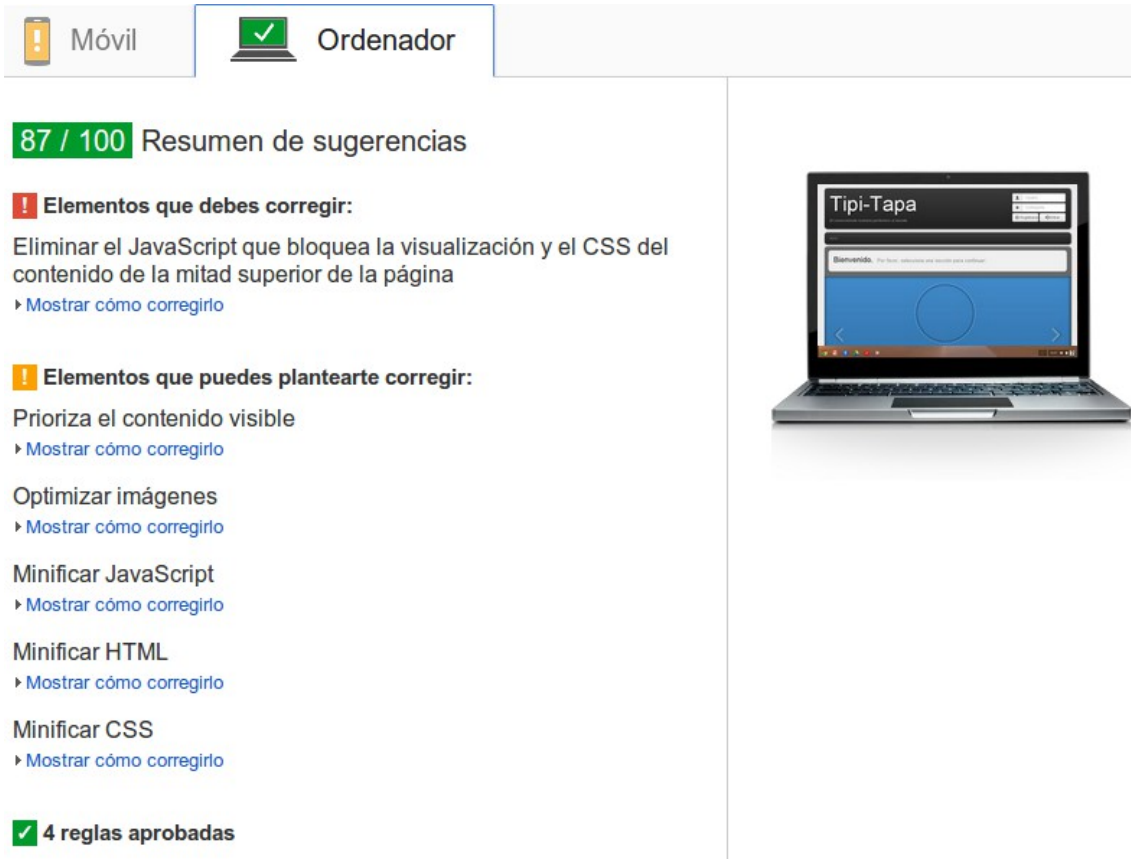


Ilustración 57: Control de optimización - Google PageSpeed Insights (ordenador)

Puesto que al tratarse de un diseño responsivo, el código ejecutado tanto en la versión móvil como en la de escritorio es el mismo, los consejos de Google no varían para este modo.

5.7.2. Pingdom Tools

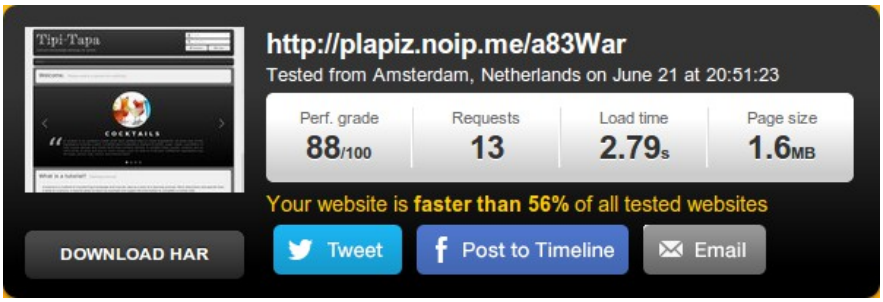


Ilustración 58: Control de optimización - Pingdom Tools (resumen)

La herramienta nos muestra la puntuación que le da a la página (notable alto) y cómo en el ranking de testeos realizados es ligeramente superior a la media aún con la tara de no tener un hosting adecuado.

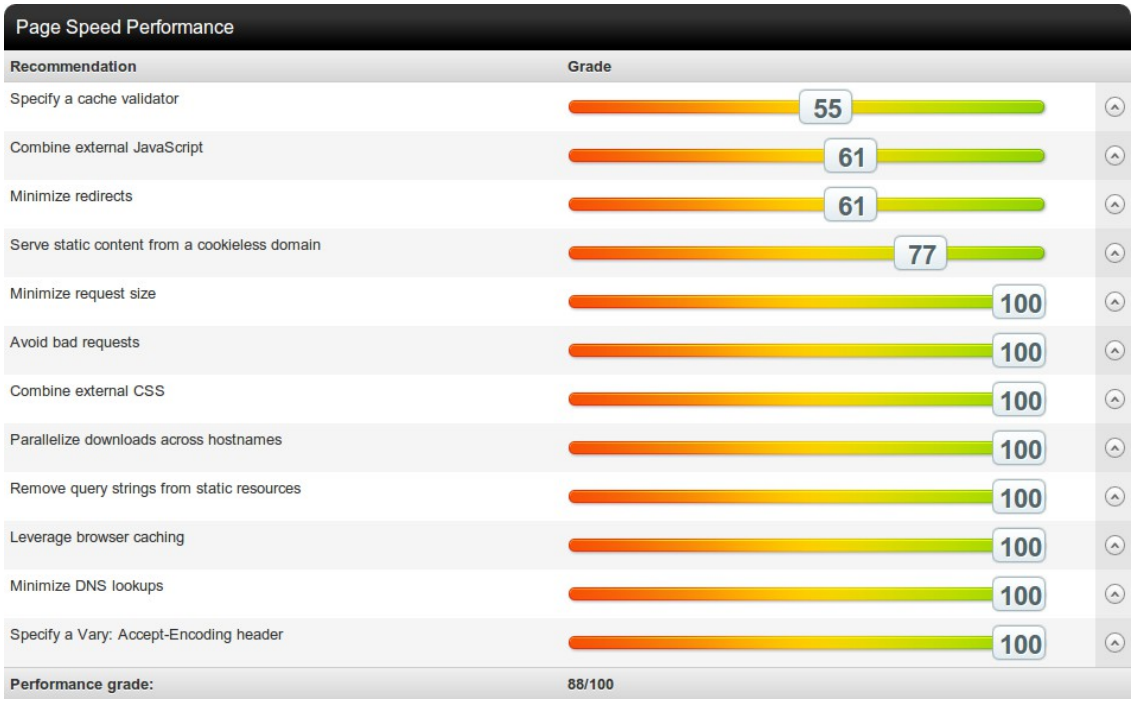


Ilustración 59: Control de optimización - Pingdom Tools (desglose)

- Combine external JavaScript:

  There are 2 JavaScript files served from plapiz.noip.me. They should be combined into as few files as possible.

  http://plapiz.noip.me/a83War/staticContent/wro/grupo.js

  http://plapiz.noip.me/a83War/staticContent/wro/validate.js

  Comenta que deberían combinarse los archivos de JavaScript para tener los mínimos posibles. Esta advertencia ya se había tenido en cuenta y se han tomado las medidas oportunas.

- Remove the following redirect chain if possible:

  http://plapiz.noip.me/a83War

  http://plapiz.noip.me/a83War/

  Alerta de la redirección de la página principal. No se tendrá en cuenta esta advertencia.

- Specify a cache validator:

  The following resources are missing a cache validator. Resources that do not specify a cache validator cannot be refreshed efficiently. Specify a Last-Modified or ETag header to enable cache validation for the following resources:

  http://plapiz.noip.me/a83War/image/seccion/220x220/2.png

http://plapiz.noip.me/a83War/image/seccion/220x220/8.png

[...]

Recomienda el uso de tags de gestión de cache en las imágenes. No se tendrá en cuenta puesto que ya se han introducido modificaciones respecto al cacheo de éstas.

- Serve static content from a cookieless domain:

  Serve the following static resources from a domain that doesn't set cookies:

  http://plapiz.noip.me/a83War/image/seccion/220x220/2.png

  [...]

  http://plapiz.noip.me/a83War/staticContent/wro/grupo.css

  http://plapiz.noip.me/a83War/staticContent/wro/grupo.js

  http://plapiz.noip.me/a83War/staticContent/wro/validate.js

  Alerta de que los contenidos estáticos debieran servirse desde un dominio que no gestione cookies. Esto no es posible puesto que las imágenes y el resto de estáticos tienen que servirse desde el servidor de aplicaciones JBOSS (y no directamente desde Apache) por las soluciones de optimización implementadas en JavaScript y CSS y el hecho de que las imágenes están almacenadas en la base de datos.

5.7.3. GTmetrix

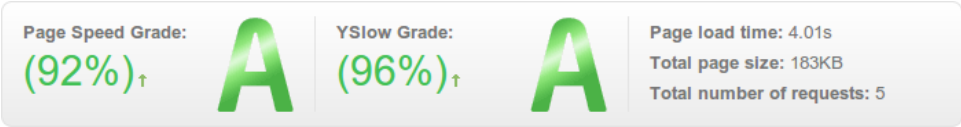
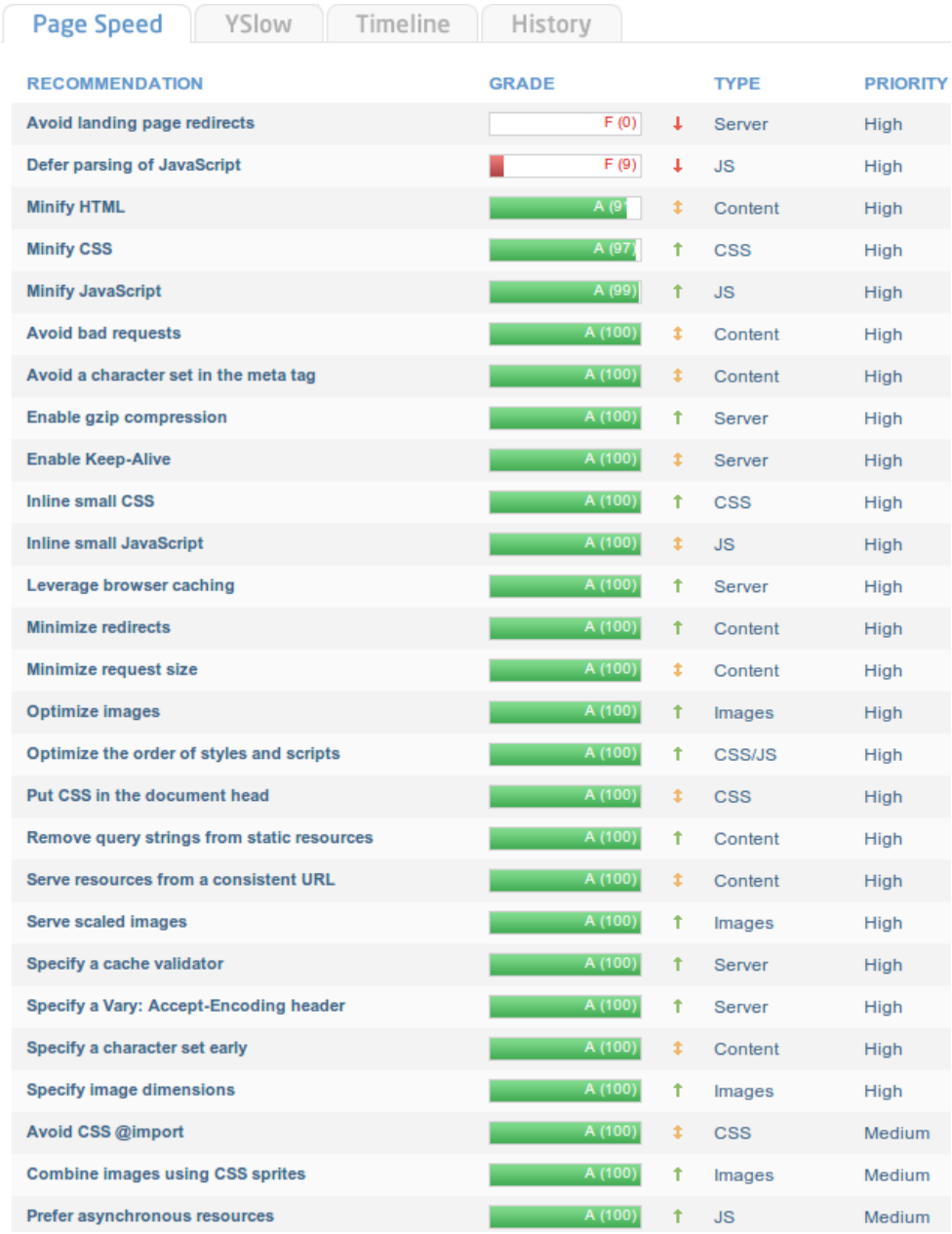


Ilustración 60: Control de optimización - GTmetrix (resumen)

Los porcentajes que nos da esta herramienta son muy altos (sobresaliente) por lo que parece que premia más la optimización de la página que los tiempos del servidor.



Page Speed | YSlow | Timeline | History

RECOMMENDATION	GRADE	TYPE	PRIORITY
Avoid landing page redirects	F (0)	Server	High
Defer parsing of JavaScript	F (9)	JS	High
Minify HTML	A (9	Content	High
Minify CSS	A (97	CSS	High
Minify JavaScript	A (99)	JS	High
Avoid bad requests	A (100)	Content	High
Avoid a character set in the meta tag	A (100)	Content	High
Enable gzip compression	A (100)	Server	High
Enable Keep-Alive	A (100)	Server	High
Inline small CSS	A (100)	CSS	High
Inline small JavaScript	A (100)	JS	High
Leverage browser caching	A (100)	Server	High
Minimize redirects	A (100)	Content	High
Minimize request size	A (100)	Content	High
Optimize images	A (100)	Images	High
Optimize the order of styles and scripts	A (100)	CSS/JS	High
Put CSS in the document head	A (100)	CSS	High
Remove query strings from static resources	A (100)	Content	High
Serve resources from a consistent URL	A (100)	Content	High
Serve scaled images	A (100)	Images	High
Specify a cache validator	A (100)	Server	High
Specify a Vary: Accept-Encoding header	A (100)	Server	High
Specify a character set early	A (100)	Content	High
Specify image dimensions	A (100)	Images	High
Avoid CSS @import	A (100)	CSS	Medium
Combine images using CSS sprites	A (100)	Images	Medium
Prefer asynchronous resources	A (100)	JS	Medium

Ilustración 61: Control de optimización - GTmetrix (page speed)

- Avoid landing page redirects:

  To speed up page load times for visitors of your site, remove as many landing page redirections as possible, and make any required redirections cacheable if possible.

  http://plapiz.noip.me/a83War is a non-cacheable redirect to http://plapiz.noip.me/a83War/

  Al igual que Pingdom Tools, alerta de la redirección de la página principal. No

se tendrá en cuenta esta advertencia.

- Minify HTML:

  Minifying the following HTML resources could reduce their size by 1.5KiB (22% reduction).

  Al igual que en la herramienta Google PageSpeed Insights alerta de que es recomendable minimizar el código HTML generado. A esta tarea se dedicarán esfuerzos en líneas futuras.

- Minify CSS:

  Minifying the following CSS resources could reduce their size by 1.1KiB (4% reduction).

  Al igual que en la herramienta Google PageSpeed Insights alerta del minimizado de CSS. Como es un porcentaje bajo dado que ya se ha minimizado el código CSS, no se tendrá en cuenta.

- Minify JavaScript:

  Minifying the following JavaScript resources could reduce their size by 58B (1% reduction).

  Al igual que en la herramienta Google PageSpeed Insights alerta del minimizado de JavaScript. Como es un porcentaje bajo dado que ya se ha minimizado el código JavaScript, no se tendrá en cuenta.

Ilustración 62: Control de optimización - GTmetrix (YSlow)

- Use a Content Delivery Network (CDN):

  Una red de entrega de contenidos (CDN) es una red superpuesta de computadoras que contienen copias de datos. Estos datos están colocados en varios puntos de una red con el fin de maximizar el ancho de banda para el acceso a los datos de clientes por la red. Un cliente accede a una copia de la información cerca del cliente, en contraposición a todos los clientes que acceden al mismo servidor central, a fin de evitar cuellos de botella cerca de ese servidor.

  No se plantean soluciones a esta alerta puesto que no habrá financiación en el proyecto.

- Use cookie-free domains:

  Al igual que en la herramienta Pingdom Tools, alerta de no tener los archivos de

estáticos en un servidor que utilice cookies. No se tendrá en cuenta.

5.8. Estado final

En este apartado se muestra el estado final de las distintas pantallas que componen la aplicación en 3 distintas resoluciones destinadas en principio a la visualización en escritorio, tableta y móvil. Para ello se emplea la herramienta on-line Screenfly (http://quirktools.com/screenfly/) para los siguientes dispositivos:

- Escritorio: 19" Desktop (1440x900).
- Tableta: Apple Ipad 1-3/Mini (768x1024).
- Móvil: Samsung Galaxy S3/4 (360x640).

Pantalla principal

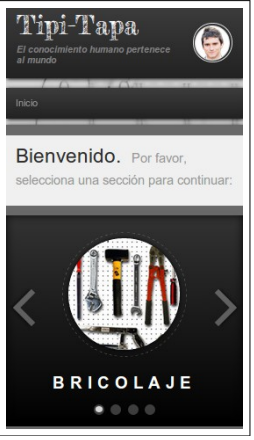
Escritorio	Tableta	Móvil
		

Tabla 39: Estado final - Pantalla principal

Presentación de tutoriales

Escritorio	Tableta	Móvil
		

Tabla 40: Estado final - Presentación de tutoriales

Ejecución de un tutorial

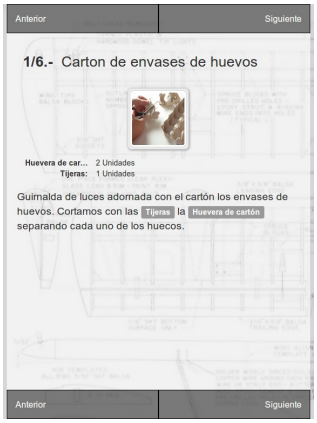
Escritorio	Tableta	Móvil
		

Tabla 41: Estado final - Ejecución de un tutorial

Administración

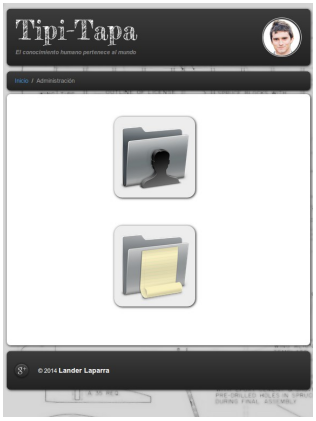
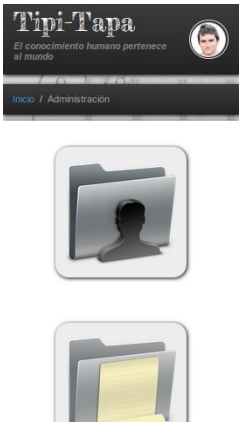
Escritorio	Tableta	Móvil
		

Tabla 42: Estado final – Administración

Gestión de usuarios

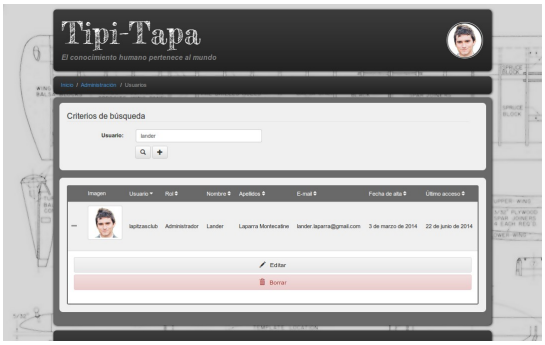
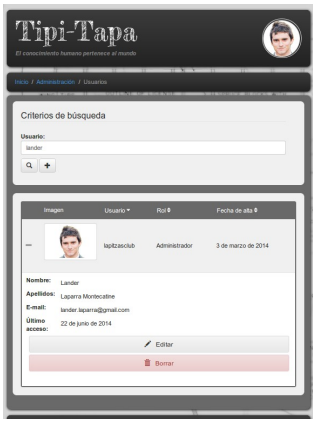
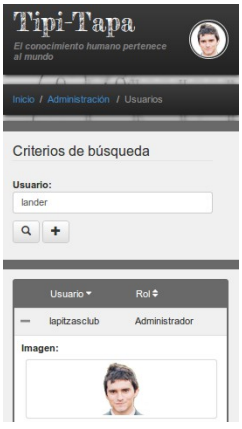
Escritorio	Tableta	Móvil
		

Tabla 43: Estado final - Gestión de usuarios

Gestión de contenidos

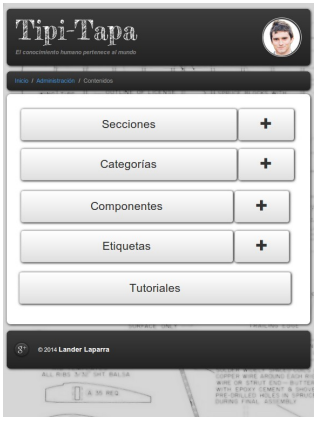
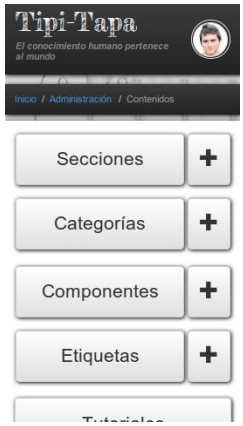
Escritorio	Tableta	Móvil
		

Tabla 44: Estado final - Gestión de contenidos

Gestión de secciones

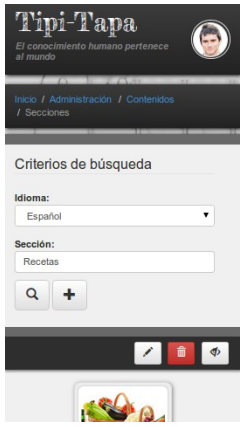
Escritorio	Tableta	Móvil
		

Tabla 45: Estado final - Gestión de secciones

Gestión de categorías

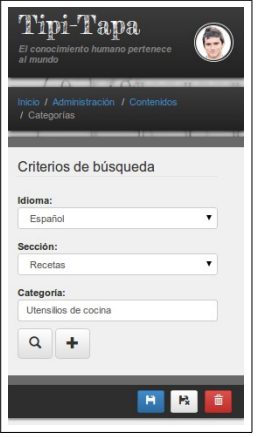
Escritorio	Tableta	Móvil
		

Tabla 46: Estado final - Gestión de categorías

Gestión de componentes

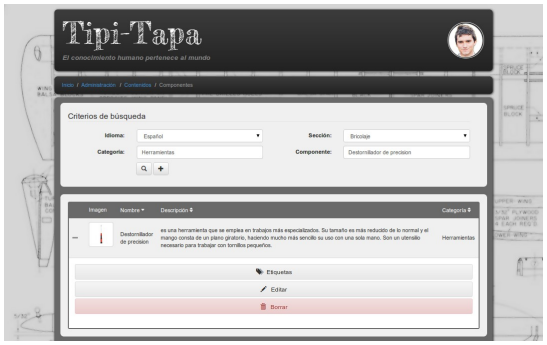
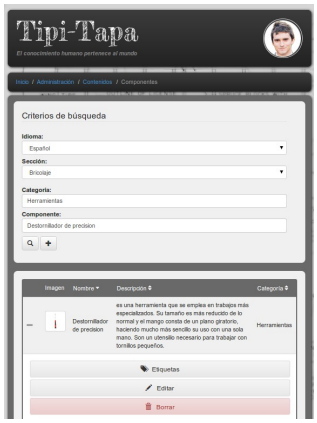
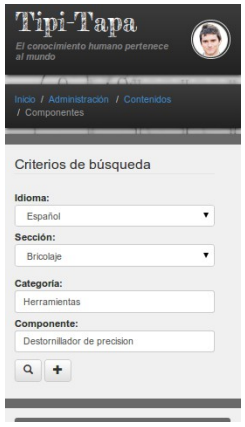
Escritorio	Tableta	Móvil
		

Tabla 47: Estado final - Gestión de componentes

Gestión de etiquetas

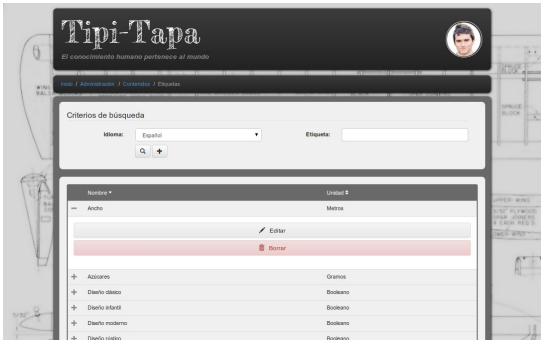
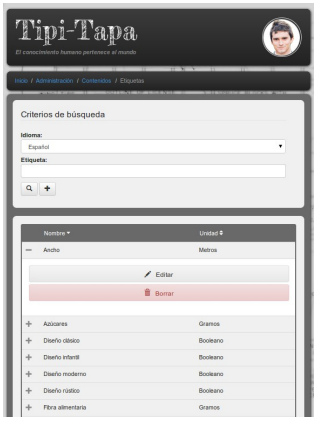
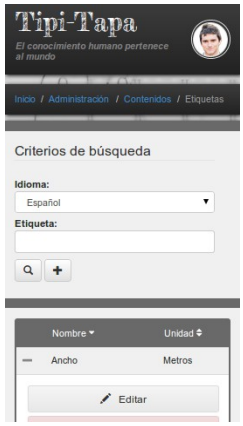
Escritorio	Tableta	Móvil
		

Tabla 48: Estado final - Gestión de etiquetas

Gestión de tutoriales

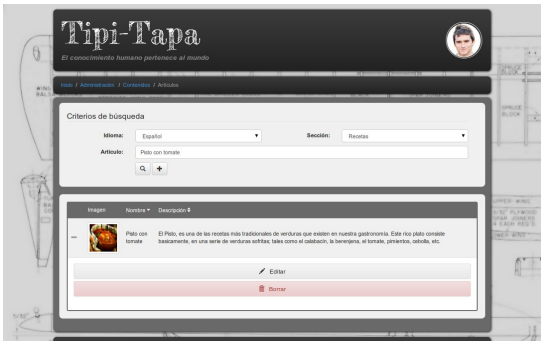
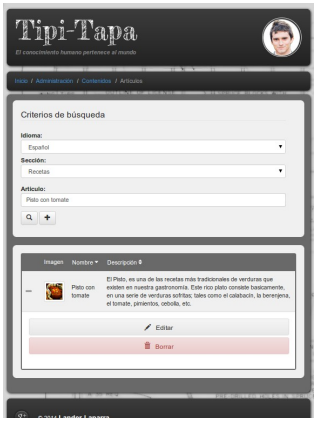
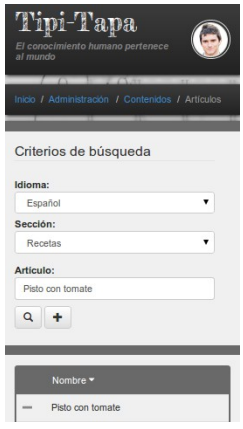
Escritorio	Tableta	Móvil
		

Tabla 49: Estado final - Gestión de tutoriales

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1. Conclusiones

La idea de este proyecto surgió por la necesidad de dar un salto tecnológico a nivel personal en las nuevas tecnologías empleadas en la construcción de aplicaciones Web.

Durante los siete años de actividad laboral en el desarrollo de aplicaciones Web, mi empresa me ha mantenido mayoritariamente en el mantenimiento de aplicaciones cuyo ciclo de vida está llegando a su fin y piden un salto generacional. Si bien es cierto que he recibido cierta formación como empleado en diferentes tecnologías, nunca he podido tener contacto con proyectos de nueva índole.

El proyecto de Tipi-Tapa ha sido capaz de darme la experiencia necesaria en el desarrollo de aplicaciones modernas con interfaces punteras.

Gracias a ella mi empresa se hizo eco. Actualmente me encuentro liderando el equipo de desarrollo de las aplicaciones que hacen uso del nuevo Framework del Gobierno Vasco, UDA (Utilidades de desarrollo de aplicaciones). Este Framework está basado en Spring, como este proyecto y para nuestras aplicaciones estamos incluyendo interfaces sensibles con Bootstrap tal y como yo lo he hecho para Tipi-Tapa.

Por otro lado creo que ha resultado un proyecto que puede tener unas expectativas de negocio más que interesantes si se sigue invirtiendo tiempo (y algo de dinero) una vez consiga un nivel de madurez más alto. Por ahora para el ámbito personal, familia y amistades cumple con su cometido.

6.2. Líneas futuras

En el futuro se espera continuar con el desarrollo del proyecto mejorando lo ya implementado o añadiéndole nuevas funcionalidades.

Entre las nuevas funcionalidades, en primer lugar habría que atender a los requisitos que no han sido atendidos por falta de tiempo. Una búsqueda más compleja de tutoriales en la parte de usuario, el poder especificar componentes no deseados en los tutoriales y el uso de cookies en la aplicación.

Otra nueva funcionalidad que podría introducirse en la aplicación sería la ampliación de la parte de administración con un apartado para visualizar estadísticas de uso con gráficos, etc. Una especie de minería de datos tan en uso hoy en día a través de programas orientados al Bussiness Inteligence.

También sería interesante plantearse un apartado en el que los usuarios sin perfil de administrador o mánager puedan sugerir o “pre-publicar” sus propios tutoriales de forma que la aplicación pueda crecer en contenidos con mayor facilidad.

Como mejoras queda pendiente el uso de herramientas de compresión de código html una vez compiladas las JSPs por el servidor de aplicaciones como parte de las cosas a mejorar en optimización.

7. ANEXOS

7.1. Análisis de las interfaces

En este apartado se muestran los bocetos de las interfaces realizados en Wireframe (guía visual que representa el esqueleto o estructura visual de un sitio web). Estos bocetos han sido realizados en base a los casos de uso especificados y desembocarán en el resultado final de la interfaz con el diseño definitivo de la misma.

Página principal vista por un usuario:

Se define la interfaz en cuatro partes:

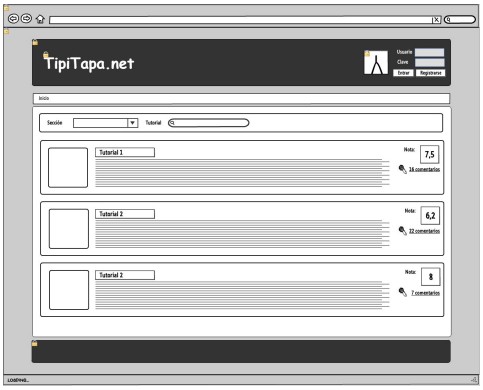


Ilustración 63: Wireframe de la página principal vista por un usuario

1. Cabecera (header)

   La cabecera contendrá el nombre de la página (Tipi-Tapa) a la izquierda y el formulario de login en la aplicación a la derecha. Este formulario de login está formado por el nombre, la contraseña y un botón de registro para darse de alta si no se tiene usuario.

2. Rastro de migas de pan (breadcrumbs)

   Contendrá los identificadores de las pantallas por las que hemos pasado y serán navegables. Podremos ir a esas pantallas al pulsar encima. En la pantalla principal pondrá Inicio.

3. Cuerpo (body)

   Se mostrará un formulario de búsqueda y los resultados encontrados de los tutoriales disponibles en la aplicación.

4. Pie (footer)

   Mostrará el copyright, el autor y enlaces varios.

Página principal vista por un usuario registrado

Una vez el usuario se loguea en la aplicación se mostrará en la cabecera la foto del usuario y el nombre. Si el usuario logueado tiene permisos de administrador/manager se mostrará un enlace "administrar" para navegar a las pantallas de administración de la aplicación. El enlace "cerrar sesión" desloguea al usuario en la aplicación volviendo a la página principal. También se visualiza un botón en la parte superior derecha para modificar el perfil del usuario.

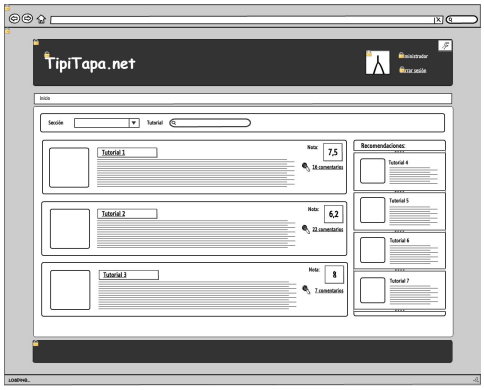


Ilustración 64: Wireframe de la página principal vista por un usuario registrado

Con el usuario logueado se mostrará contenido exclusivo al usuario registrado, valoraciones de otros usuarios, recomendaciones, etc.

Detalle de un tutorial

En esta pantalla se muestra información detallada del tutorial seleccionado. Además los usuarios registrados podrán valorar y comentar el artículo. Contiene información como:

- El título.
- Una breve descripción.
- Listado de los componentes necesarios.
- El tutorial en sí formado por la conjunción de todos sus pasos uno detrás de otro.
- Valoraciones de otros usuarios.
- Un botón para ejecutar el tutorial paso por paso.

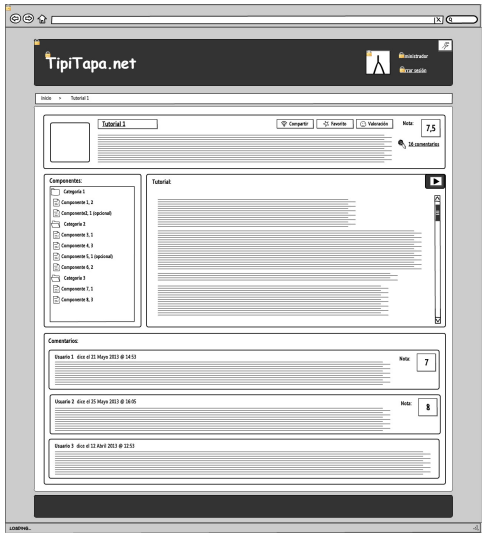


Ilustración 65: Wireframe del detalle de un tutorial

Ejecución de un tutorial

A la hora de ejecutar un tutorial se mostrará una pantalla como esta, donde se muestra información relevante del primero de los pasos que componen el tutorial ejecutado.

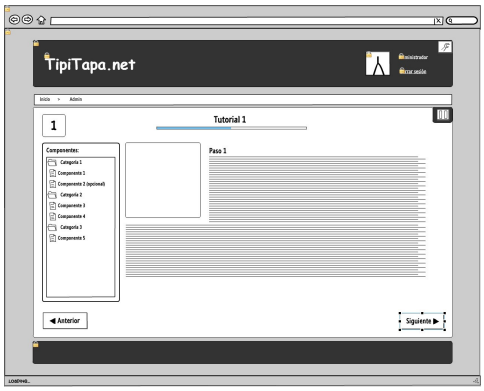


Ilustración 66: Wireframe de la ejecución de un tutorial

La información que se muestra es la siguiente:

1. El título del tutorial.
2. El número de paso en el que estamos.
3. Una barra que indique el progreso dentro de los pasos.
4. La lista de componentes necesarios para ejecutar el paso.
5. Una foto del paso.
6. La descripción de lo que se debe hacer en el paso.
7. Un botón para salir a la pantalla anterior.
8. Botones de atrás y siguiente para navegar entre pasos del tutorial.

Pantalla de administración

Esta es la pantalla a la que acceden los administradores al pulsar el botón "Administrar" de la cabecera. Los managers al pulsar en "Administrar" accederán directamente a la pantalla de administración de contenido. Si por algún casual accediesen por URL a esta pantalla no tendrán bloqueo.
En esta pantalla se mostrarán dos grandes botones:

- Gestión de usuarios (sólo visible por el administrador).
- Gestión de contenidos.

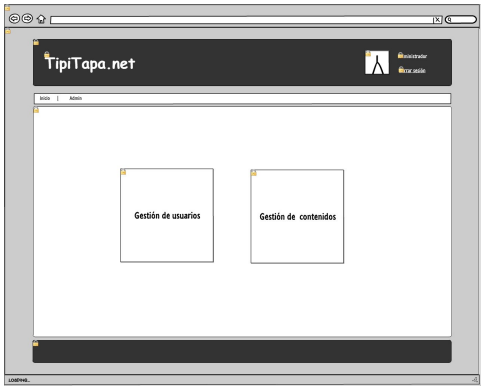


Ilustración 67: Wireframe de la pantalla de administración

Gestión de usuarios

En esta pantalla se mostrará un formulario de búsqueda que permitirá buscar usuarios por: su rol en la aplicación (usuario, manager o administrador), su estado (activo o bloqueado) o el nombre.

Al pulsar en buscar se lanzará la búsqueda y se cargará la tabla con los resultados coincidentes.

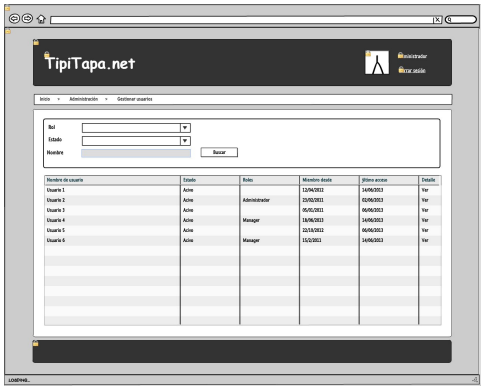


Ilustración 68: Wireframe de la gestión de usuarios

Edición de usuarios

En la pantalla de detalle de un usuario se mostrará su foto y un formulario con sus datos personales. Tendremos tres botones: uno para cancelar la edición, otro para guardar y otro para activar o bloquear al usuario en función del estado en el que se encuentre.

Adicionalmente se visualizarán estadísticas del usuario en el manejo de la aplicación.

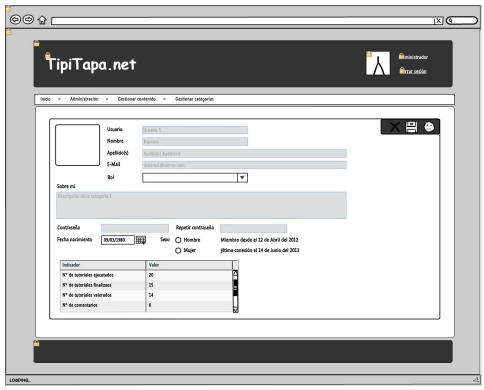


Ilustración 69: Wireframe de edición de usuarios

Gestión de contenidos

Desde la pantalla de gestión de contenidos se mostrarán varios botones grandes:

- Gestión de secciones: Navega a la pantalla de gestión de secciones.
- Nueva sección: Abre un formulario para insertar una nueva sección.
- Gestión de categorías: Navega a la pantalla de gestión de categorías.
- Nueva categoría: Abre un formulario para insertar una nueva categoría.

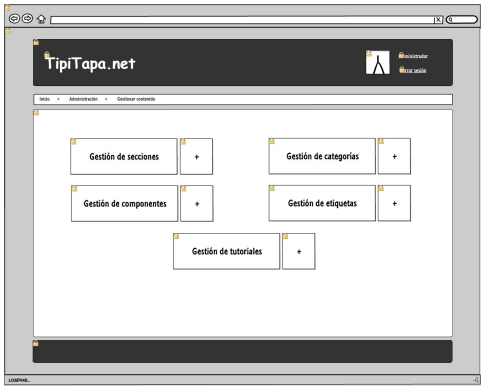


Ilustración 70: Wireframe de gestión de contenidos

- Gestión de componentes: Navega a la pantalla de gestión de componentes.
- Nuevo componente: Abre un formulario para insertar un nuevo componente.
- Gestión de etiquetas: Navega a la pantalla de gestión de etiquetas.
- Nueva etiqueta: Abre un formulario para insertar una nueva etiqueta.
- Gestión de tutoriales: Navega a la pantalla de gestión de tutoriales.
- Nuevo tutorial: Navega a la pantalla de creación de tutoriales.

Gestión de contenidos - Nueva sección

Se mostrará un formulario de alta con el nombre, la descripción y la foto de la sección. Esta pantalla será similar para el alta de secciones y componentes.

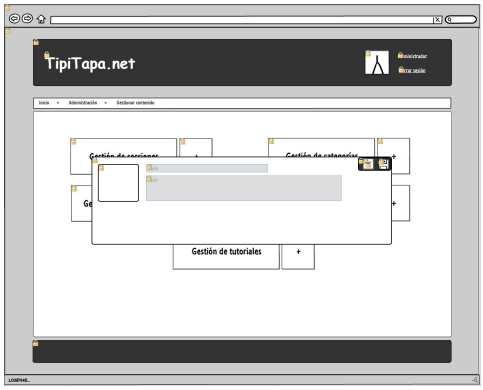


Ilustración 71: Wireframe de Gestión de contenidos - Nueva sección

Gestión de contenidos - Nueva etiqueta

Se mostrará un formulario de alta con el nombre, la descripción, la foto y las unidades de medida de la etiqueta.

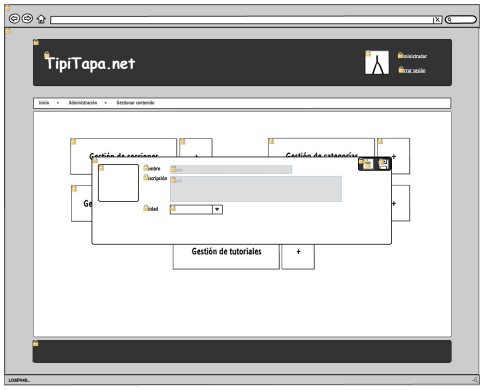


Ilustración 72: Wireframe de gestión de contenidos - Nueva etiqueta

Gestión de secciones

En esta pantalla se muestra la información pertinente a la gestión de secciones. Desde aquí se podrán consultar y modificar las secciones ya creadas y componer otras nuevas.

Las secciones que aparezcan en los resultados de búsqueda tendrán tres botones: uno para eliminar, otro para editar y otro para publicar.

Al editar una sección se habilitarán los campos y aparecerá un botón para guardar los cambios y otro para cancelarlos.

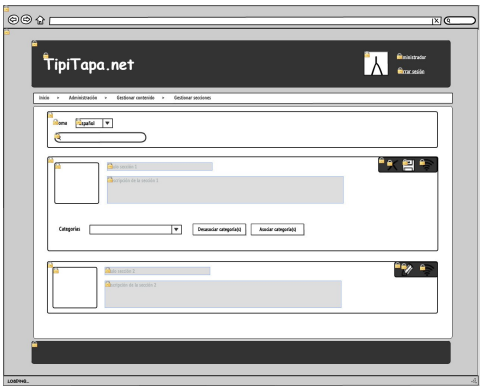


Ilustración 73: Wireframe de gestión de secciones

Gestión de categorías

Esta pantalla es estructuralmente similar a la de gestión de secciones. En ella se muestra la información pertinente a la gestión de categorías. Desde aquí se podrán consultar y modificar las categorías ya creadas y componer otras nuevas.

Las categorías que aparezcan en los resultados de búsqueda tendrán tres botones: uno para eliminar, otro para editar y otro para publicar.

Al editar una categoría se habilitarán los campos y aparecerá un botón para guardar los cambios y otro para cancelarlos.

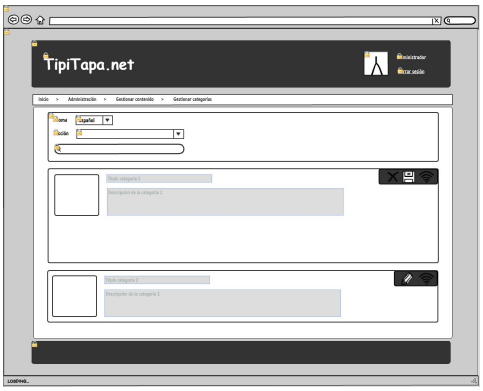


Ilustración 74: Wireframe de gestión de categorías

Gestión de componentes

En esta pantalla el diseño es distinto a las anteriores puesto que el número de componentes a mantener en la aplicación será sensiblemente mayor de los que se guardarán para secciones o categorías.

En esta pantalla se muestra un filtro de búsqueda y una tabla de resultados de componentes a los que podremos navegar para consultar o editar.

También podremos hacer altas de nuevos componentes desde esta pantalla.

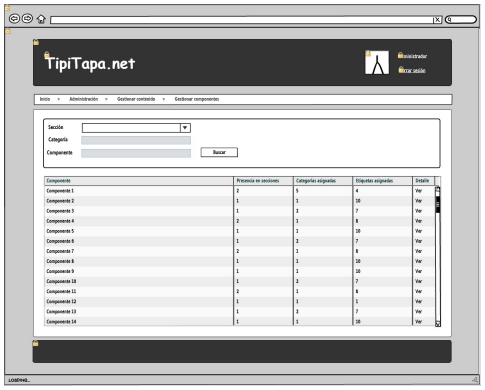


Ilustración 75: Wireframe de gestión de componentes

Edición de componentes

Desde esta pantalla podremos modificar los datos de un componente. Como su nombre, descripción, categoría, etiquetas asociadas, etc.

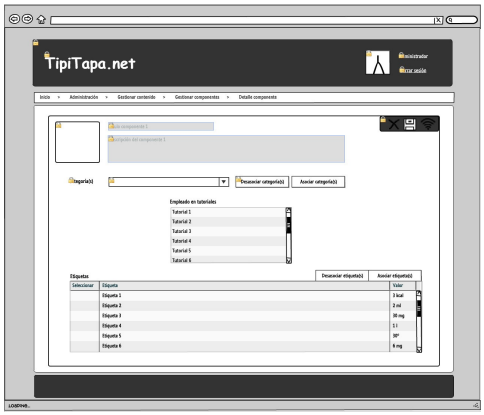


Ilustración 76: Wireframe de edición de componentes

Gestión de etiquetas

Esta pantalla será similar a la gestión de componentes. Se visualiza un formulario de búsqueda de etiquetas y su tabla de resultados donde se podrá eliminarlas o navegar para consultar/editarlas.

Desde esta pantalla también se permitirá crear nuevas etiquetas.

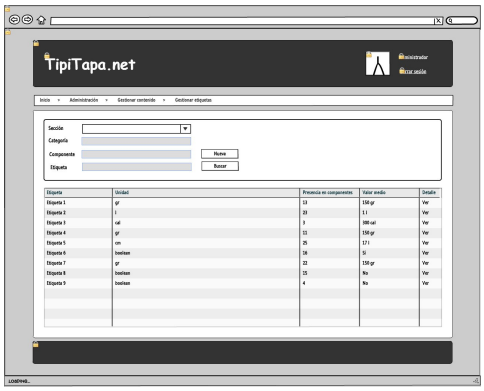


Ilustración 77: Wireframe de gestión de etiquetas

Nueva etiqueta

Al dar de alta una nueva etiqueta se mostrará el formulario de alta de etiquetas.

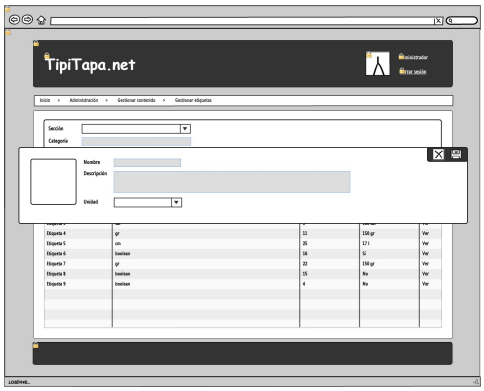


Ilustración 78: Wireframe de nueva etiqueta

Edición de etiquetas

Desde esta pantalla se podrán modificar los datos de la etiqueta como el nombre, la descripción, unidad de medida, etc.

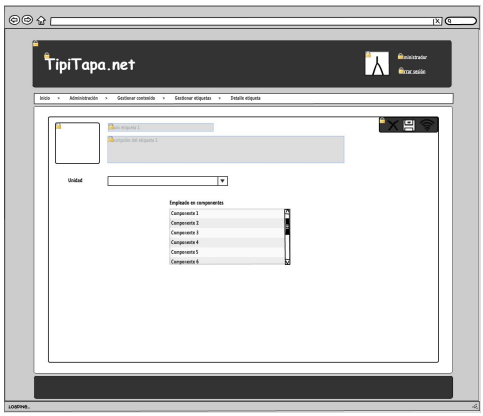


Ilustración 79: Wireframe de edición de etiquetas

Gestión de tutoriales

Ésta es una de las pantallas más importantes de la aplicación (o quizás la más importante) junto con la pantalla de visualización de los tutoriales, puesto que desde aquí se van a administrar los contenidos que visualizarán los usuarios.

La primera pantalla de gestión de tutoriales muestra, al igual que las anteriores, un filtro de búsqueda y una tabla de resultados con los componentes filtrados. Por cada uno de los resultamos se tendrá la opción de eliminarlo o navegar a su pantalla de detalle/modificación. También dispondrá de un botón para dar de alta nuevos componentes.

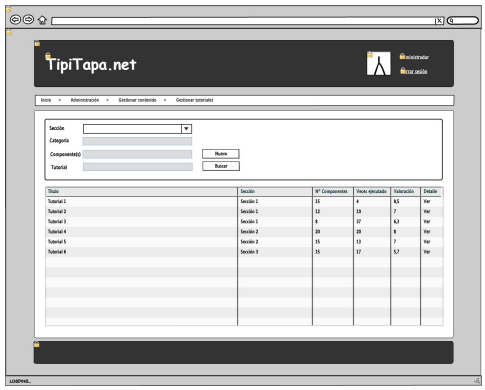


Ilustración 80: Wireframe de gestión de tutoriales

Nuevo tutorial

Al dar de alta un nuevo componente se abrirá un formulario con los datos necesarios para crear un nuevo componente: su sección, el nombre, descripción, etc.

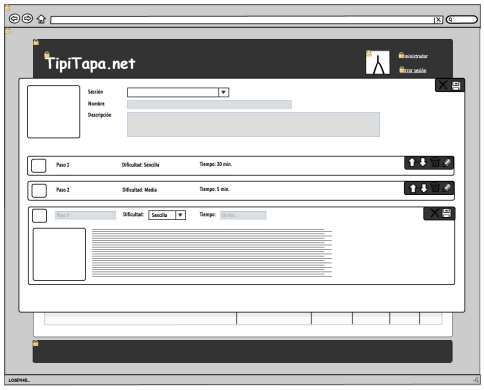


Ilustración 81: Wireframe de nuevo tutorial

Editar tutoriales - Datos básicos

En el detalle de un componente tendremos sus datos desglosados en 3 pestañas. La primera de ellas contiene la información básica del tutorial como son su sección, el nombre, la descripción o la foto.

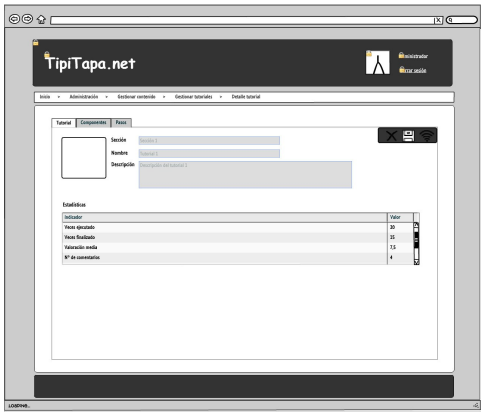


Ilustración 82: Wireframe de editar tutoriales - Datos básicos

Editar tutoriales - Componentes

En la segunda pestaña del detalle de un tutorial se visualizan los componentes empleados en él con sus cantidades, etc.

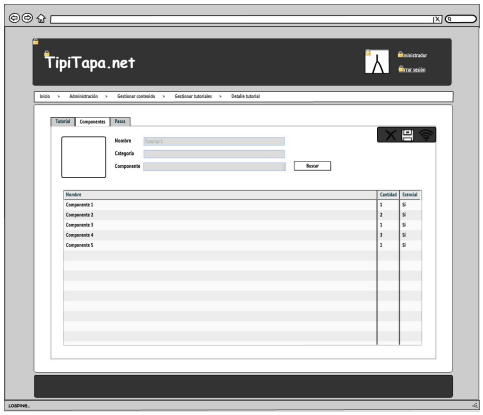


Ilustración 83: Wireframe de editar tutoriales - Componentes

Editar tutoriales - Pasos

En la pestaña de pasos del detalle de tutoriales se presentan los pasos de los que consta el tutorial para poder modificarlos o crear pasos nuevos.

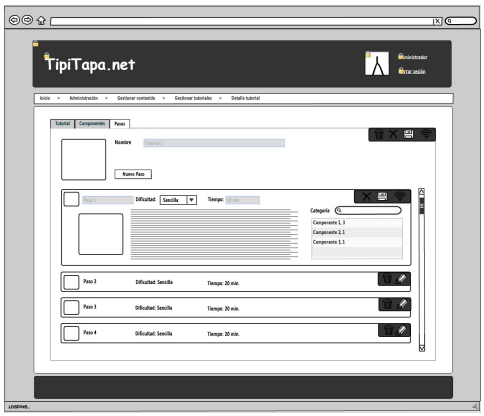


Ilustración 84: Wireframe de editar tutoriales - Pasos

8. BIBLIOGRAFÍA

[1] Guía Ubuntu, comunidad hispana de ubuntu, http://www.guia-ubuntu.com/

[2] Softonic, enorme repositorio de aplicaciones, http://www.softonic.com

[3] Idearium, marketing y comunicación, http://www.idearium30.com/

[4] Apache, https://httpd.apache.org/

[5] JBOSS Web Server, http://www.jboss.org/products/webserver/overview/

[6] StarUML, http://staruml.sourceforge.net/

[7] Eclipse, plataforma y plugins, http://www.eclipse.org/

[8] jQuery, API y jQueryUI, http://jquery.com/

[9] Spring, http://spring.io/

[10] Bootstrap, http://getbootstrap.com/

[11] TouchSwipe, https://github.com/mattbryson/TouchSwipe-Jquery-Plugin

[12] BlockUI, https://github.com/malsup/blockui/

[13] SlitSlider, http://tympanus.net/codrops/2012/06/05/fullscreen-slit-slider-with-jquery-and-css3/

[14] Fit Text, https://github.com/davatron5000/FitText.js

[15] FooTable, http://themergency.com/footable/

[16] Bootstrap Dialog, http://nakupanda.github.io/bootstrap3-dialog/

[17] Bootstrap MaxLength, http://mimo84.github.io/bootstrap-maxlength/

[18] jQuery Mentions, http://podio.github.io/jquery-mentions-input/

[19] jQuery Share In1, http://plugins.in1.com/share

[20] Bootstrap wizard, https://github.com/VinceG/twitter-bootstrap-wizard

8.1. Foros

[1] La Web del programador, http://www.lawebdelprogramador.com/

[2] Stack Ovreflow, http://stackoverflow.com/

8.2. Libros

[1] Van Lancker, Luc. *HTML5 y CSS3. Domine los estándares de las aplicaciones*

Web. Ed. eni.

[2] Van Lancker, Luc. jQuery. El Framework JavaScript de la Web 2.0. Ed. eni.

[3] Deléglise, Didier. *MySQL 5 (versiones 5.1 a 5.6). Guía de referencia del desarrollador.* Ed. eni.