**ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN**

**GRADO EN INGENIERÍA INFORMÁTICA EN TECNOLOGÍAS DE LA INFORMACIÓN**

**ÁREA DE CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIAL**

**TRABAJO FIN DE GRADO Nº 17010059**

**MEJORA DEL RENDIMIENTO Y AMPLIACIÓN DE UNA WEB PARA LA GESTIÓN INTEGRAL DE CONGRESOS CIENTÍFICOS**

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

# 

# DOCUMENTO 1: MEMORIA

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

1.1 Resumen

El presente trabajo fin de grado consiste en la mejora del rendimiento y ampliación de una web ya existente y en producción, dedicada a la gestión integral de congresos científicos. En particular, la implementación realizada tiene como objeto la gestión del congreso *International Conference on Computational and Mathematical Methods in Science and Engineering* (de aquí en adelante CMMSE.

Parte de la aplicación web ha sido desarrollada por un antiguo alumno de la escuela politécnica de ingeniería de Gijón. Dicha aplicación lleva en uso desde 2014, y ha sustituido a otra aplicación previa de objetivo similar. Cientos de profesionales científicos han usado la aplicación hasta día de hoy.

En primer lugar, se han actualizado las tecnologías existentes en el proyecto para optimizar la seguridad y el rendimiento tanto del servidor como de la aplicación. En segundo lugar, se ha mejorado la interfaz de usuario y la usabilidad de la aplicación, además de añadir nuevas funcionalidades a petición de la directiva del CMMSE.

La aplicación web funcionará principalmente como una herramienta de gestión integral de congresos científicos, celebrados una vez al año. Los usuarios serán capaces de registrarse para acudir al congreso anual, además de poder subir archivos a la plataforma con contenido científico para presentarlos en dicho congreso. También podrán seguir el proceso de aceptación de artículos a presentar, y el alta y modificación de sus datos y archivos. Otra de las partes importantes de la aplicación es la del usuario administrador, ya que será el único usuario capaz de ver, modificar o borrar los datos relativos a los usuarios registrados, relacionados al congreso y de configuración de la aplicación.

La herramienta principal que se ha escogido para el desarrollo de esta aplicación web ha sido el framework web *open source* Angular de Google. Más en concreto su versión 5, sacada al mercado el pasado 1 de noviembre de 2017, sustituyendo a su versión anterior Angular 4. Lenguajes como HTML5, Sass, TypesScript, JavaScript o jQuery se han utilizado para satisfacer las necesidades del *front-end* del proyecto, garantizando la mejor experiencia de usuario posible. Por otro lado, la base de datos utilizada para almacenar y gestionar la información de la aplicación web es Firebase Cloud Firestore, también de Google. Por parte del servidor, el lenguaje de programación usado es Node.js por lo que la aplicación ha sido desarrollada a medida para ejecutarse sobre un servidor web o para ser subida a un hosting con unos requisitos mínimos, teniendo en cuenta las particularidades del congreso a gestionar, el presupuesto y del entorno *hardware* y *software* disponible. Otro punto importante a destacar es el uso de GitHub, repositorio de control de versiones usado a lo largo del desarrollo del proyecto.

1.2 Introducción

Este trabajo tiene como objetivo la ampliación de desarrollo y mejora de una aplicación web para la gestión de una conferencia científico-técnica sobre la aplicación de métodos matemáticos en problemas de ciencia e ingeniería que se celebra desde el año 2001 ininterrumpidamente. La aplicación desarrollada tiene como finalidad la gestión completa del congreso y su administración.

El proyecto surge con el objetivo de mejorar una aplicación web existente que la organización venía utilizando en ediciones anteriores del congreso. Para ello se ha realizado un estudio de las funcionalidades cubiertas por la aplicación existente en la actualidad que se encuentra en producción, además de considerar e implementar numerosas mejoras relativas a la usabilidad, a los tiempos de respuesta de la aplicación y al uso de componentes *software* actuales, utilizando librerías de código abierto *(open source)* ampliamente testadas por diferentes comunidades de desarrolladores de aplicaciones.

Inicialmente se plantea la continuación de la aplicación existente ya que la aplicación funciona correctamente a excepción de algunos errores encontrados y mejoras mínimas. Por ello, se realiza durante los primeros meses de desarrollo un mantenimiento y solución de *bugs* para que la edición de 2017 pudiese llevarse a cabo. Gracias a esto, durante las fechas de edición del congreso de ese mismo año, la aplicación estuvo operativa satisfaciendo las necesidades de los asistentes y la directiva.

Posteriormente, se llegó a un punto de no retorno por lo que la aplicación fue reconstruida por completo. Las nuevas funcionalidades requeridas y la actualización de las tecnologías supusieron la necesidad de rehacer la aplicación desde cero. Por supuesto, la nueva aplicación mantendrá los requisitos existentes.

La nueva aplicación desarrollada cubre todos los aspectos principales de la gestión de la conferencia, desde el registro de los asistentes y el envío de los artículos a presentar, hasta la generación de facturas, justificantes de pago, asistencia y presentación de ponencias, además del envío de notificaciones por correo electrónico tanto a los asistentes como a los gestores de la misma. También cubre otros aspectos administrativos, como la gestión de documentos oficiales del congreso, la generación de listados de usuarios y artículos, la verificación de pagos o la gestión de aspectos logísticos, como la preparación de etiquetas identificativas de distintos tipos para los asistentes y el personal.

En este primer capítulo correspondiente a la memoria del proyecto se plantean en primer lugar los objetivos que se pretenden conseguir con su realización y su alcance. A continuación, se expondrán los estudios y análisis realizados, así como la organización general de documentos que sigue este trabajo fin de grado. Acto seguido se detallarán las conclusiones y ampliaciones tras haber realizado el trabajo de desarrollo del sistema y finalmente se expondrá la bibliografía consultada.

1.3 Objetivos y alcance

A continuación, se expondrán los principales objetivos que se pretenden alcanzar con el desarrollo de este trabajo fin de grado, así como el alcance de éste.

Se dividirá este apartado en dos, mostrando en primer lugar los objetivos principales de la aplicación que se va a desarrollar, los cuales detallarán el servicio que se pretende proporcionar a aquellos usuarios que utilicen la aplicación. En segundo lugar, se mostrarán aquellos objetivos de ámbito académico que se pretenden conseguir durante el desarrollo de este trabajo.

1.3.1 Objetivos de la aplicación

El principal objetivo de este trabajo fin de grado es desarrollar una ampliación de una plataforma web dirigida a las personas que asistan al CMMSE y a todas las personas que la dirigen.

De esta forma los usuarios finales podrán usar la plataforma online con todos los requisitos de seguridad necesarios y con las últimas actualizaciones de todas las tecnologías que se usan en la actualidad. Su interfaz será mucho más intuitiva permitiendo una navegación más cómoda para todos los usuarios. El aspecto contará con una apariencia moderna respetando las peticiones del CMMSE.

La usabilidad de la plataforma será sencilla, así como práctica, ofreciendo al conferenciante la posibilidad de inscribirse al congreso y administrar todos sus datos y archivos de manera online, sin necesidad de realizar ninguna gestión presencial.

Por otro lado, la dirección del CMMSE utilizará la aplicación para gestionar todo el congreso de forma online. Desde ella podrá consultar rápidamente cualquier dato de los usuarios, de los artículos que se van a presentar o de las facturas generadas, entre otras muchas cosas.

La aplicación web será accesible a través de Internet, tanto para conferenciantes como para el personal del CMMSE, accediendo por medio de un navegador o bien descargando la aplicación desde el repositorio de GitHub en el que se encuentra alojada.

1.3.2 Objetivos académicos y de aprendizaje

Otro de los objetivos principales de este trabajo fin de grado es desarrollar las nuevas funcionalidades de la aplicación con los lenguajes de programación actuales destinados al desarrollo web.

Debido al crecimiento de este tipo de aplicaciones en los últimos años, el conocimiento y uso de este tipo de lenguajes se ha convertido en una de las competencias profesionales más demandadas e importantes en el sector de la programación web hoy en día y por tanto de la informática.

Además, se pretende cumplir con unos objetivos mínimos de seguridad en el servidor de producción y en las tecnologías usadas para el desarrollo de la aplicación.

Algunos de los objetivos académicos y de aprendizaje que se pretenden conseguir con el desarrollo de este trabajo fin de grado son:

* Desarrollar una ampliación de funcionalidades de una aplicación web ya existente, con los lenguajes y tecnologías más modernos del mercado.
* Utilizar el patrón de diseño arquitectónico modelo-vista-controlador (MVC) durante el desarrollo de las nuevas funcionalidades de la aplicación.
* Utilizar un sistema de gestión de bases de datos moderno y escalable con acceso en tiempo real.
* Mejorar la usabilidad y accesibilidad de la aplicación actual, utilizando herramientas de desarrollo web de gran importancia en la actualidad, así como lenguajes de estilos para la interfaz de la aplicación.
* Utilizar una plataforma de control de versiones durante el desarrollo.
* Aplicar las técnicas necesarias para blindar el servidor de producción y la aplicación ante cualquier posible ataque o uso inapropiado del sistema.

1.4 Estudios y análisis previos

A continuación, se presentarán aquellas plataformas, tecnologías o herramientas más modernas, potentes y utilizadas hoy en día en el mercado de las aplicaciones web. Se dividirá este apartado en dos; estudios de carácter teórico y estudios de carácter técnico.

1.4.1 Estudios de carácter teórico

En los últimos años el desarrollo de aplicaciones web se ha incrementado notablemente, por lo que la mayoría de las aplicaciones que se pueden encontrar en la web han sido mejoradas y actualizadas. Entre ellas, también se incluyen aquellas plataformas web dedicadas a la gestión de congresos científico-tecnológicos. Cada vez son más las personas que utilizan estas plataformas con el objetivo de asistir a congresos internacionales y así gestionar la asistencia a dichos congresos desde su lugar de residencia.

Plataformas como Aconf, EasyChair, o Primoris ofertan diversos servicios para gestionar eventos o conferencias como la CMMSE. Entre sus servicios está el alojamiento de la aplicación, la gestión de la base de datos, su desarrollo y despliegue y su personalización.

En este tipo de aplicaciones web las organizaciones se registran y contratan un servicio que en algunos casos puede ser gratuito. Se trata de un contrato de software para gestionar conferencias, con el objetivo de ahorrarse el desarrollo de la aplicación por parte de la organización del congreso. Sin embargo, esto tiene la desventaja de no poder ofrecer a los usuarios de la aplicación todas las funcionalidades deseadas o totalmente adaptadas a lo que se necesita. La organización contratante deberá ajustarse al servicio contratado.

Por otro lado, estas plataformas se ofertan en algunos casos gratuitamente con unos servicios mínimos, con la idea de atraer clientes que más tarde necesitarán alguno de los servicios de pago. Es el caso de Aconf y EasyChair.

Otra de las posibilidades existentes en el mercado es el desarrollo de la aplicación de gestión del congreso por parte de la propia organización que lo imparte. Un ejemplo es Sistedes, congreso científico técnico de Ingeniería y Tecnologías del Software.

A continuación, se hará una breve descripción de las plataformas anteriormente mencionadas, así como las herramientas con las que se han desarrollado.

1.4.1.1 Aconf (<https://www.aconf.org>)

Creada oficialmente en el año 2013, esta plataforma es gratuita para congresos académicos. Se trata de un *service as a software* (*Saas*), diseñado para mejorar la eficiencia de la gestión de congresos y conferencias antes, durante y después de su ejecución.

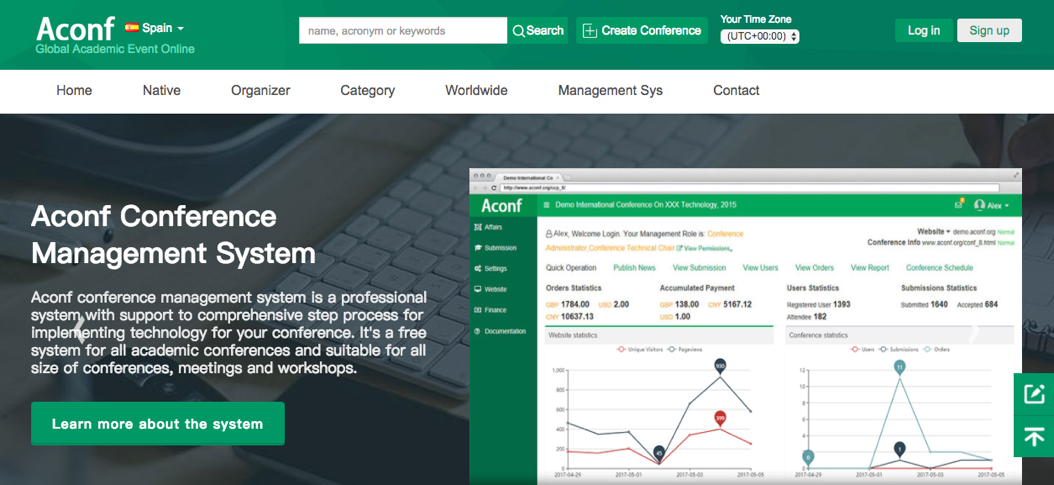


Figura 1 Página inicial de Aconf

Respecto a las tecnologías utilizadas, está basado en una herramienta WYSIWYG (*What You See Is What You Get*). Se trata de un sistema de gestión de contenidos (CMS) que permite escribir un documento viendo directamente el resultado final, es decir, se genera una vista preliminar del código HTML que luego el programa se encarga de generar con su código fuente HTML. En cuanto al resto de tecnologías, las mantienen en secreto.

Entre las ventajas de esta aplicación está la creación de plataformas de gestión de congresos académicos totalmente gratuitos. Su principal objetivo es promover la comunicación académica y construir comunidades académicas.

Una de las desventajas obvias es que no permiten la creación de plataformas no dedicadas a propósitos académicos.

1.4.1.2 EasyChair ([www.easychair.org](http://www.easychair.org))

Junto a Aconf, EasyChair es otro de los portales web más famosos para gestionar congresos. Se encuentra en el mercado desde el 2002, con más de dos millones de usuarios y más de 60.000 conferencias en sus servidores. Al igual que Aconf, oferta un servicio gratuito para conferencias académicas. En cambio, en su web ofertan un servicio mejor para aquellos que necesiten más prestaciones.

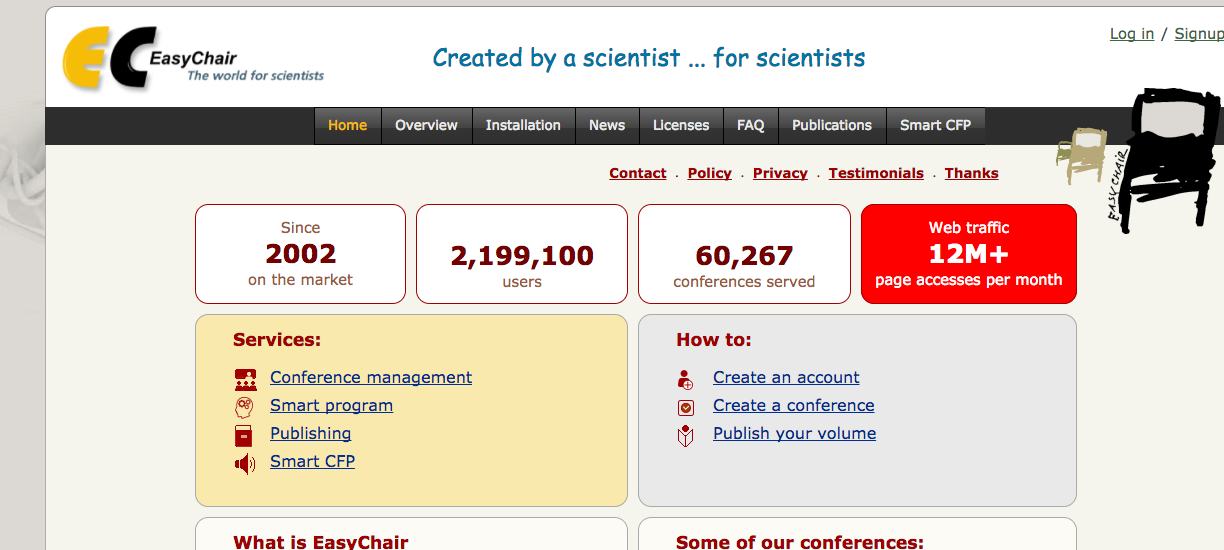


Figura 2 Página inical de EasyChair

El lenguaje para el desarrollo de esta plataforma es PHP. Al igual que Aconf, el resto de tecnologías usadas se mantienen en secreto.

En relación con Aconf, el propósito de sus servicios es similar, con la diferencia de que EasyChair ofrece un servicio de creación de aplicación para aquellas organizaciones no académicas que deseen organizar un congreso.

1.4.1.3 Primoris ([www.insticc.org](https://www.insticc.org))

Esta plataforma que se comercializa con el slogan “*La primera de la nueva generación de sistemas de gestión de eventos*”, soporta todo el proceso de gestión de congresos como Aconf y EasyChair. Con una interfaz moderna, gestiona todas las operaciones necesarias para el *front-end* y *back-end* de la aplicación. Organizada en módulos, tiene diferentes alternativas de configuración y personalización.

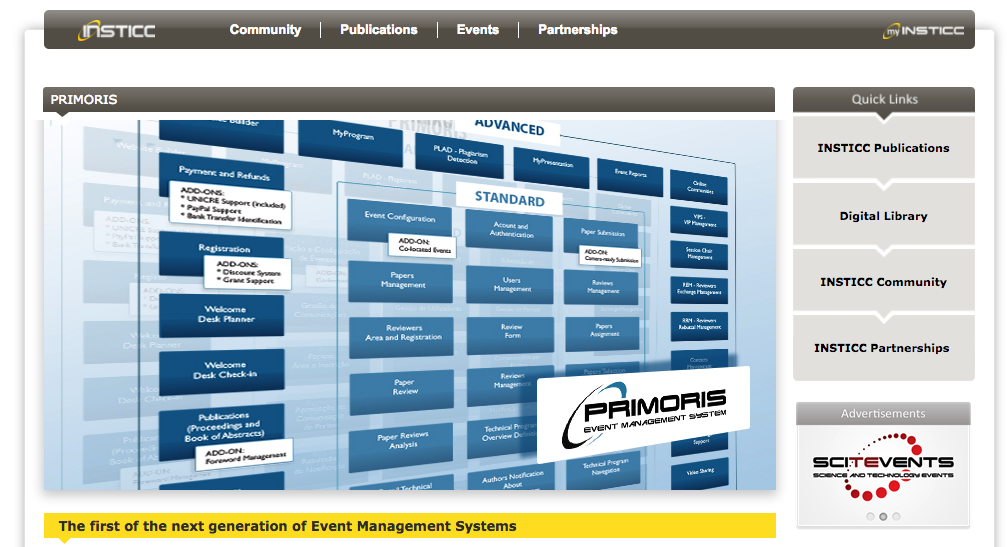


Figura 3 Página inicial de Primoris

La plataforma ha sido desarrollada mediante el entorno de aplicaciones web ASP.NET usando el *framework* .NET de Microsoft.

La diferencia con las otras plataformas ya comentadas es que Primoris no ofrece públicamente sus tarifas. Es por ello por lo que para contratar uno de sus servicios hay que entrar en contacto con el departamento de eventos. Dos de los congresos que utilizan esta plataforma son:

* <http://www.icsoft.org/>
* <http://www.enase.org/>

1.4.1.4 Sistedes (<https://fg.ull.es/sistedes2017>)

Esta plataforma perteneciente a la Universidad de La Laguna de Tenerife fue desarrollada con el objetivo de organizar las jornadas de la Sociedad de Ingeniería del Software y Tecnologías de Desarrollo de Software (SISTEDES). Este evento científico-técnico de carácter nacional engloba diferentes conferencias nacionales sobre ingeniería del software, bases de datos, ciencia e ingeniería de servicios y programación y lenguajes.



Figura 4 Página inicial de Sistedes

La principal diferencia con el resto de las plataformas es que está gestionada únicamente para la misma organización, por lo que las tecnologías seleccionadas para su desarrollo son las óptimas para la gestión de las conferencias. En cambio, su desarrollo es más largo.

En cuanto a las tecnologías, Sistedes fue desarrollada con WordPress, un sistema de gestión de contenidos (CMS) muy utilizado a día de hoy. WordPress está basado en PHP y MySQL ejecutado en un servidor Apache. En el lado del cliente, WordPress proporciona un amplio abanico de plantillas creadas por diversos desarrolladores. Estas plantillas suelen estar hechas con código JavaScript, jQuery, HTML5 y CSS3. Algunas de estas plantillas son gratuitas pero la mayoría son de pago.

Sin duda, Sistedes es una de las plataformas web más parecidas al tipo de aplicación que se desea desarrollar en este trabajo fin de grado, ya que su finalidad es ajustar las funcionalidades deseadas por el CMMSE al desarrollo, de modo que no haya restricción a la hora de personalizar la interfaz y la lógica de la aplicación.

Tras analizar varias de las plataformas web más conocidas y usadas actualmente para la gestión de congresos, se puede llegar a la conclusión de que una de las mejoras respecto a otras aplicaciones similares será la usabilidad de la aplicación. Para ello se mejorará la interfaz existente y se implementará un estilo definido para toda la aplicación que ayudará a mejorar la experiencia de usuario.

Una vez expuestas las plataformas anteriormente mencionadas, se hará una recopilación de las mejores y más populares herramientas usadas en la actualidad para desarrollar una aplicación web, ya sean lenguajes de programación web, *frameworks* o lenguajes de bases de datos.

1.4.2 Estudios de carácter técnico

En primer lugar, se diferenciará entre *framework* y lenguaje de programación. En general, un *framework* es aquel entorno o estructura software que cuenta con aquellos componentes necesarios para realizar una aplicación, un *framework* podrá estar desarrollado por uno o más lenguajes de programación distintos y normalmente están estructurados con patrones de diseño. Estos componentes del *framework* se encargan de facilitar la tarea al desarrollador ya que agilizan el desarrollo de una aplicación.

Por otra parte, los lenguajes de programación son utilizados por los desarrolladores para realizar aquellas acciones que se desean ejecutar, es decir, es el lenguaje que interactúa directamente con la máquina.

Por último, se hará hincapié en los gestores de bases de datos, parte fundamental de cualquier sistema informático, encargados del almacenamiento de la información y de su tratamiento.

A continuación, se expondrán más detalladamente estos conceptos.

1.4.2.1 Patrones de diseño

En *frameworks* de ámbito web, es imprescindible utilizar uno de los patrones de arquitectura de software más conocidos, el patrón MVC (modelo-vista-controlador), patrón de diseño por el cual se rigen la mayoría de *frameworks* de desarrollo web.

Como se puede ver en la [Figura 5](#Figura5), la principal característica del patrón MVC es que está basado en la división de capas y/o conceptos, organizando la aplicación en tres modelos separados: modelo, vista y controlador.

1.4.2.1.1 Modelo

Esta capa se encarga de comunicarse con la base de datos mediante las correspondientes peticiones que hace un usuario, pudiendo acceder a los datos, así como también modificarlos, eliminarlos o crearlos si es preciso.

1.4.2.1.2 Vista

Capa encargada de tratar el código de la interfaz de usuario para presentar los datos que el modelo proporciona. Es lo que el usuario verá de la aplicación. Para aplicaciones web actuales, los lenguajes de programación más usados en el mercado y que están estrechamente ligados a la vista son HTML y CSS.

1.4.2.1.3 Controlador

Esta es una de las capas más importantes, ya que sirve de intermediario entre la vista y el modelo. A diferencia del modelo, el controlador no accede a los datos ni los manipula, si no que envía las correspondientes peticiones originadas por los usuarios al modelo cuando estos solicitan información y también se encarga de enviar los respectivos comandos a las vistas asociadas a estos cuando se solicita un cambio en la forma que estas se presentan.

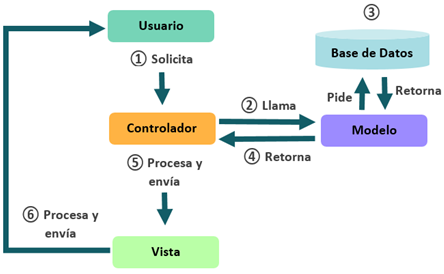


Figura 5 Esquema patrón MVC

Entre las ventajas de utilizar *frameworks* de desarrollo web que utilicen este patrón de diseño se encuentran las siguientes:

* La rapidez con que se desarrolla la aplicación, dado que esta se divide en módulos diferentes e independientes, los cuales se pueden desarrollar en cualquier momento con independencia de los demás.
* Se integra fácilmente con lenguajes de programación web como PHP, JavaScript, HTML, jQuery, etc. y se puede compatibilizar con técnicas de comunicación asíncrona (como AJAX).
* La facilidad para crear múltiples vistas para un modelo. Cada una de ellas será independiente del resto de vistas de la aplicación.
* La organización de código entre las diferentes capas ayuda a una sencilla implementación de nuevas funcionalidades dentro de la aplicación desarrollada.

1.4.2.2 Lenguajes de Programación

Los lenguajes de programación en el ámbito del desarrollo web son aquellos lenguajes que un navegador web podrá entender y, por lo tanto, para los que está preparado para interactuar permitiendo que el usuario visualice desde el navegador un sistema o aplicación que se encuentra alojado en una máquina dentro de la red. Normalmente en un proyecto de desarrollo web se clasifican los lenguajes en dos tipos, lenguajes en el lado del servidor (*back-end*) y lenguajes en el lado del cliente (*front-end*).

Los lenguajes en el lado del servidor (*back-end*) son aquellos lenguajes que se ejecutan en el servidor y se encargan de interactuar con la base de datos, así como también crear las sesiones de usuario o proporcionar los datos necesarios para las vistas de la aplicación. Lenguajes como PHP, Java, Ruby, Python, C# o Node.js son algunos de los lenguajes más utilizados en la actualidad en el lado del servidor.

Por otra parte, los lenguajes en el lado del cliente (*front-end*) son aquellos lenguajes o tecnologías que se ejecutan en el propio navegador web. Los ficheros escritos en estos lenguajes son servidos directamente al navegador para que los interprete. Los navegadores web tienen la capacidad de maquetar y estilizar la vista con la que el usuario está interactuando, con el objetivo de mejorar la experiencia que el usuario tiene con la aplicación. Lenguajes como HTML5, CSS3, JavaScript, TypeScript o librerías como jQuery y/o *frameworks* como Angular o React son los principales lenguajes del lado del cliente usados en la mayoría de aplicaciones web.

Estos dos tipos de lenguajes no tienen mucho éxito el uno sin el otro, ya que el *back-end* de una aplicación necesitará de los lenguajes del lado del cliente para que el usuario pueda interactuar con la información de la base de datos y el *front-end* necesitará aquella información suministrada por el *back-end* para poder mostrar los resultados de sus interacciones.

A continuación, se listan algunos de los diferentes lenguajes *back-end* con sus correspondientes *frameworks*, los cuales siguen el patrón de diseño MVC y que son muy utilizados en el mercado actualmente:

1.4.2.2.1 PHP

Para este lenguaje los *frameworks* más utilizados son Laravel, CakePHP, Symfony o CodeIgniter.

1.4.2.2.2 Java

Struts, Spring o JavaServerFaces son algunos de los *frameworks* más usados.

1.4.2.2.3 Ruby

Para Ruby el *framework* más usado es Ruby on Rails.

1.4.2.2.4 Python

Para este lenguaje de programación el *framework* más usado en la actualidad es Django.

1.4.2.2.5 C#

El *framework* usado por este lenguaje es .NET, propiedad de Microsoft.

1.4.2.2.6 Node.js

Basado en JavaScript, los *frameworks* más potentes y utilizados son Angular y React.

1.4.2.3 Sistema de gestión de bases de datos

Para finalizar este apartado, una de las partes más importantes en una aplicación web es el sistema de gestión de bases de datos, que almacenará aquella información esencial de la aplicación web de la forma más eficiente y segura posible y que deberá estar disponible en el momento que se necesite.

Normalmente en el desarrollo de aplicaciones web las bases de datos suelen ser relacionales (generalmente SQL ya que es el lenguaje más extendido). Sin embargo, en la actualidad muchos desarrolladores optan por una nueva tendencia, las bases de datos no relacionales. A continuación, se exponen algunas de las características más importantes de estos dos tipos:

* Las bases de datos relacionales permiten combinar de una forma eficiente diferentes tablas con el objetivo de extraer información relacionada, mientras que las no relacionales no lo permiten o lo hacen con un peor rendimiento que las relacionales. Ejemplos: PostgreSQL, MySQL, Oracle, etc.
* Las bases de datos no relacionales permiten distribuir grandes cantidades de información de forma más rápida, por lo que este tipo de bases de datos están pensadas para el uso en redes sociales o destinadas al Big Data, es decir, cuando el volumen de datos es de gran tamaño o las estructuras de datos son variables. También cuentan con un escalado horizontal, es decir, al agregar más nodos su rendimiento mejora considerablemente. Ejemplos: MongoDB, Firebase Cloud Firestore, Cassandra, CouchDB, Redis, etc.

Esta breve introducción al estudio de carácter técnico de este trabajo fin de grado no tiene como objetivo debatir cuál de las tecnologías anteriormente indicadas es mejor. Sin embargo, es importante exponer cuáles son las ventajas e inconvenientes de cada una de ellas. El objetivo final es escoger la tecnología adecuada para la realización de la aplicación web.

En el apartado “3.1.4. Análisis de alternativas” del documento número 3 “Requisitos de usuario y análisis de alternativas” se hará una descripción más detallada de algunos de los lenguajes y herramientas anteriormente mencionados, aportando datos y resultados reales, para posteriormente escoger las tecnologías con las que se desarrollará la aplicación web.

1.5 Organización del documento

En este apartado se mostrará una lista detallada de los diferentes capítulos que componen este trabajo fin de grado, además de una breve descripción de cada uno de ellos.

1.5.1 Documento 1 – Memoria

Se trata del presente capítulo, en él se incluye en primer lugar un resumen del proyecto y una introducción. A continuación, los objetivos y el alcance de este trabajo fin de grado. Después se detallan los estudios y análisis previamente realizados. Por último, se comentarán las conclusiones a las que se ha llegado al finalizar el presente trabajo y se mostrará la organización de los diferentes capítulos que componen este trabajo fin de grado, además de la bibliografía consultada.

1.5.2 Documento 2 – Presupuesto y planificación

En este capítulo se detallará la planificación temporal de las diferentes fases del proyecto, desde el inicio hasta su conclusión. A continuación, se describirá el cálculo del coste total del proyecto, así como el coste de los recursos necesarios para el desarrollo de este.

1.5.3 Documento 3 – Requisitos de usuario y análisis de alternativas

En este documento se incluirá la especificación de requisitos de usuario y análisis de alternativas de este trabajo fin de grado. Para acabar el documento se expondrá la alternativa seleccionada.

1.5.4 Documento 4 – Análisis de requisitos del sistema

En este capítulo se recogerán los casos de uso de toda la aplicación, así como también se detallarán los requisitos no funcionales y el modelo de datos.

1.5.5 Documento 5 – Diseño del sistema

En este capítulo se expondrán los diagramas de paquetes, de clases, de iteración, así como también el diseño de la base de datos y el diseño de la interfaz.

1.5.6 Documento 6 – Pruebas

En este documento se detallarán las pruebas que se han hecho durante el desarrollo de este: pruebas unitarias, de integración y sistema, de usabilidad y accesibilidad y de rendimiento.

1.5.8 Documento 7 – Manuales

En este último documento se detalla una descripción detallada de los pasos que hay que seguir para instalar la aplicación desde cero, además de una recopilación de las librerías más importantes utilizadas en el desarrollo de la aplicación web.

1.6 Conclusiones y ampliaciones

En el presente trabajo fin de grado se han definido claramente los objetivos y el alcance del mismo, por lo que en este apartado se hablará de las conclusiones a las que se ha llegado, además de una pequeña lista de las posibles ampliaciones que pueden llevarse a cabo en esta aplicación.

1.6.1 Conclusiones

Finalizado todo el proceso de implementación y desarrollo asociado a este trabajo fin de grado se puede llegar a la conclusión de que se ha cumplido con los objetivos propuestos inicialmente.

Se han desarrollado una serie de funcionalidades nuevas en la aplicación web y ha sido mejorada la experiencia de usuario para los dos posibles roles definidos (asistentes del congreso y personal de la organización), utilizando tecnologías actualizadas y modernas en el campo de las tecnologías web.

Por otro lado, se han actualizado las tecnologías que se usan en la aplicación web, aumentando la seguridad de la aplicación y su consistencia. Además, la aplicación es escalable en cuanto a cantidad de datos posibles a almacenar.

En definitiva, este trabajo fin de grado ha sido de gran utilidad para la organización ya que se ha conseguido mejorar considerablemente la aplicación y su seguridad con un presupuesto limitado.

1.6.2 Ampliaciones

Con el desarrollo de la aplicación actual finalizado, no cabe duda de que aún existen algunas mejoras que se podrían llevar a cabo. Cualquier aplicación o sistema puede ser mejorado y/o ampliado continuamente, por lo que a continuación se detallan algunas de las posibles ampliaciones y mejoras. Entre ellas se encuentran mejoras de las tecnologías, nuevas funcionalidades, mantenimiento continuo del sistema y mejoras de seguridad.

1.6.2.1 Mantenimiento continuo de actualizaciones de *software*

El software implementado en esta aplicación proviene de librerías, lenguajes y sistemas operativos de código abierto. Esta comunidad *open source* es muy amplia por lo que continuamente se renueva el software utilizado y todas sus versiones. Lo ideal para un sistema de estas características es la actualización continua de todas las versiones estables que salen al mercado. De esta forma se mejora el rendimiento y la seguridad, ya que estas actualizaciones mejoran ambos aspectos.

1.6.2.2 Creación de aplicación móvil nativa

Una de las posibles mejoras que más fundamento tiene es la creación de una aplicación móvil nativa para Android e iOS del sistema. Una de las grandes ventajas de usar Angular es que se adapta con mucha facilidad a la responsividad de las pantallas móviles. Además, existen algunos *frameworks* como Ionic, que permiten adaptar la mayoría del código web de una aplicación en Angular para convertirla con unos simples comandos de consola a una aplicación móvil nativa.

Por otro lado, otra ventaja muy importante es el uso de Firebase. Los datos de la base de datos se encuentran alojados en la nube y gracias a su arquitectura, los datos pueden ser servidos en tiempo real tanto a aplicaciones web como a aplicaciones nativas móvil Android e iOS.

1.6.2.3 Internacionalización

La aplicación cuenta con una internacionalización mínima, puede seleccionarse inglés o español como idioma dentro de la aplicación. Sin embargo, muchos de los usuarios de la aplicación tienen otra lengua nativa.

Esta mejora se basa en la inclusión de otros idiomas en la internacionalización, además de modificaciones para idiomas como el árabe, que se leen en modo espejo respecto a los idiomas occidentales como el español y por lo que habría que realizar cambios en la interfaz para que los textos y campos del sistema se pudieran leer de derecha a izquierda en lugar de izquierda a derecha.

1.6.2.4 Creación de un Chatbot en el sistema

Una de las tendencias actuales en el mercado web es el uso de un Chatbot dentro de las aplicaciones web. Se trata de un componente que ayude al usuario a realizar acciones dentro del sistema con un agente de IA que responda a las preguntas o cuestiones que el usuario se plantea. Dialogflow de Google es una de las plataformas que más valor tienen a día de hoy. Con esta herramienta se pueden crear agentes que interaccionen con el usuario si este lo desea.

1.7 Bibliografía

A continuación, se incluye la bibliografía consultada en la realización de este trabajo fin de grado. En primer lugar, se hará referencia a los libros y/o artículos. En el siguiente punto, se detallarán los sitios web consultados.

1.7.1 Libros y artículos

Durante el desarrollo del proyecto se han consultado los siguientes libros, ya sea en versión papel o en versión digital:

* **Guía de PHP**: Diego Lázaro – Leanpub, 2015.
* **Ng-book: The Complete Guide to Angular 4**: Nathan Murray – FULLSTACK.io, 2017.
* **Angular: From Theory To Practice**: Asim Hussain – Code Craft , 2016.
* **JavaScript and JQuery: Interactive Front-End Web Development**: Jon Duckett – Wiley, 2014.
* **Firebase Cookbook**: Houssem Yahiaoui – Packt, 2017.

1.7.2 Referencias en internet

En este apartado se detallarán los sitios webs consultados para el desarrollo del proyecto, así como para la documentación y la instalación de las herramientas utilizadas:

* **Angular**: <https://angular.io/>
* **Angular Material**: <https://material.angular.io/>
* **Node.js**: <https://nodejs.org/es/>
* **Firebase**: <https://firebase.google.com/>
* **Visual Studio Code**: <https://code.visualstudio.com/>
* **Sass**: <https://sass-lang.com/>
* **Medium**: <https://medium.com/>
* **Express Server**: <http://expressjs.com/es/>
* **NPM Package manager**: <https://www.npmjs.com/>
* **Udemy – Cursos online**: <https://www.udemy.com/>
* **Librería jsPDF**: <https://parall.ax/products/jspdf>
* **Plantillas HTML y Bootstrap**: <https://startbootstrap.com/>
* **Programa virtualización de máquinas VirtualBox**: <https://www.virtualbox.org/>
* **Sistema operativo CentOS**:<https://www.centos.org/>
* **Servidor Apache**:<https://httpd.apache.org/>
* **Lenguaje PHP**: <http://php.net/>
* **Doctrine ORM**: <http://www.doctrine-project.org/>
* **Tutoriales web W3Schools**: <https://www.w3schools.com/>
* **Lenguaje de bases de datos MySQL**: <https://www.mysql.com/>
* **Librería Bootstrap**: <http://getbootstrap.com>
* **Librería jQuery**: <https://jquery.com/>
* **Lenguaje JavaScript**: <https://www.javascript.com/>
* **Librería Datatables**: <https://datatables.net/>
* **GitHub – Control de versiones**: <https://github.com/>
* **Comunidad de desarrolladores**: <https://es.stackoverflow.com/>
* **Información, definiciones y significados**: <https://www.wikipedia.org/>
* **Artículos y blogging sobre infraestructura web**: <https://geekflare.com/>
* **Congreso Sistedes**: <https://fg.ull.es/sistedes2017/>
* **Congreso ICSOFT**: <http://www.icsoft.org/>
* **Congreso ENASE**: <http://www.enase.org/>
* **Sistema de gestión de congresos (Aconf)**: <https://www.aconf.org/>
* **Sistema de gestión de congresos (EasyChair)**: <http://easychair.org/licenses.cgi>
* **Sistema de gestión de congresos (Primoris)**: <https://www.insticc.org/Primoris/Default.aspx/>
* **Comunidad *open source* de seguridad del *software***: <https://www.owasp.org/>
* **Plataforma de *cloud computing* con comunidad de soporte**: <https://www.digitalocean.com>
* **Artículo de usabilidad web**: <https://www.crazyegg.com/blog/principles-website-usability/>
* **Artículo de métodos de usabilidad**: <http://www.usabilityfirst.com/usability-methods/>
* **Herramienta de medición de usabilidad**: <http://www.tawdis.net>

# DOCUMENTO 2: PLANIFICACIÓN Y PRESUPUESTO

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

## 2.1 Introducción

El presente documento se dividirá en dos partes, siendo la primera la planificación temporal del desarrollo de este trabajo fin de grado y la segunda parte una estimación detallada del coste total de la realización de este.

2.2 Planificación temporal

El inicio de este proyecto comenzó el 16 de noviembre de 2016, con fecha prevista de finalización para el 15 de mayo de 2018, sumando una duración total de 18 meses aproximadamente. El desarrollo se ha producido con una hora al día de media, sin incluir festivos ni fines de semana o periodos vacacionales.

El desarrollo principal se ha dividido en cuatro fases principales: fase de análisis, fase de diseño, fase de implementación y fase de pruebas. Además, se ha planificado también la parte correspondiente a la documentación. A continuación, se detallan distintas imágenes pertenecientes al diagrama Gantt creado durante el desarrollo del proyecto, con el objetivo de establecer una planificación detallada y concisa del trabajo.

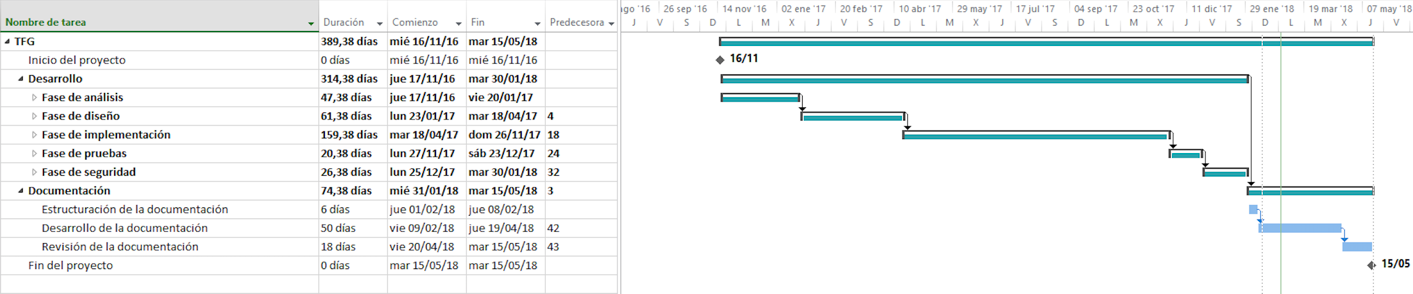


Figura 6 Diagrama Gantt – Visión global

2.2.1 Fase de análisis

Se trata de la fase de análisis del proyecto. En primer lugar, se detalla la planificación de los estudios y análisis previos. Más tarde los requisitos de usuario y el análisis de alternativas. Por último, se planifica el análisis de requisitos del sistema.

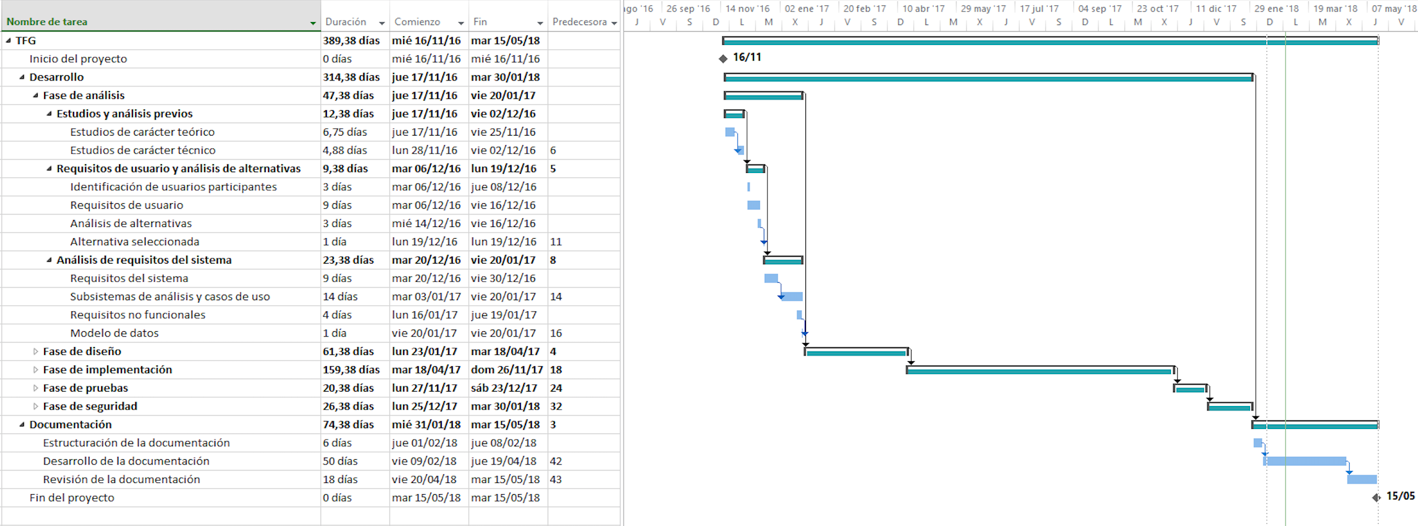


Figura 7 Diagrama Gantt - Fase de análisis

2.2.2 Fase de diseño

En esta fase se planifican los diseños específicos de ingeniería del software, diseño de la base de datos y de la interfaz de usuario.

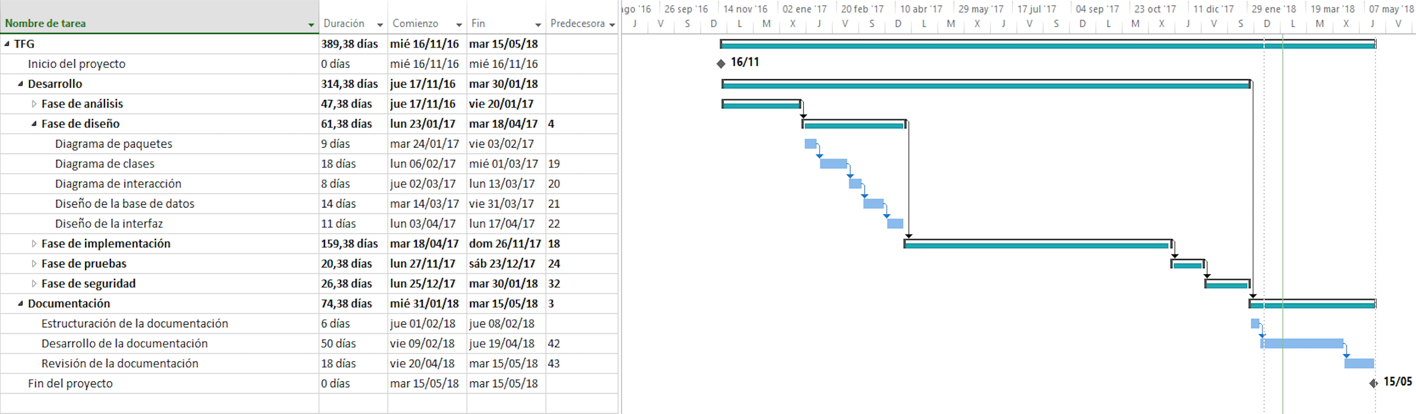


Figura 8 Diagrama Gantt - Fase de diseño

2.2.3 Fase de implementación

Es la fase de creación de la aplicación y de su estructura, y de programación de todas las funcionalidades necesarias en la mejora de la web.

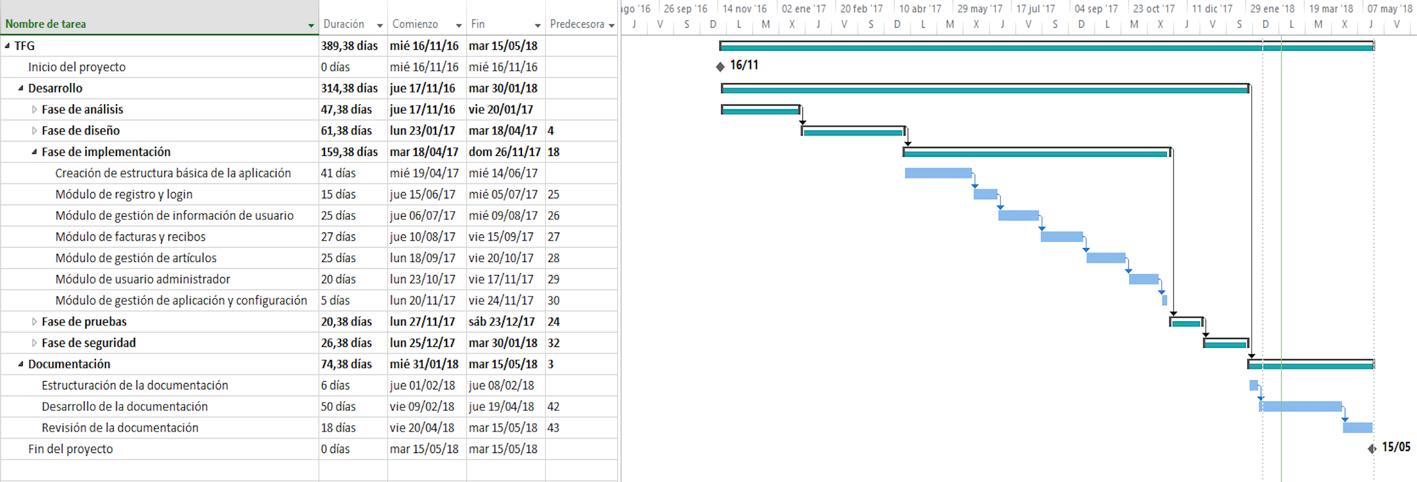


Figura 9 Diagrama Gantt - Fase de implementación

2.2.4 Fase de pruebas

Se trata de la fase de pruebas de la aplicación. Pruebas funcionales y no funcionales.

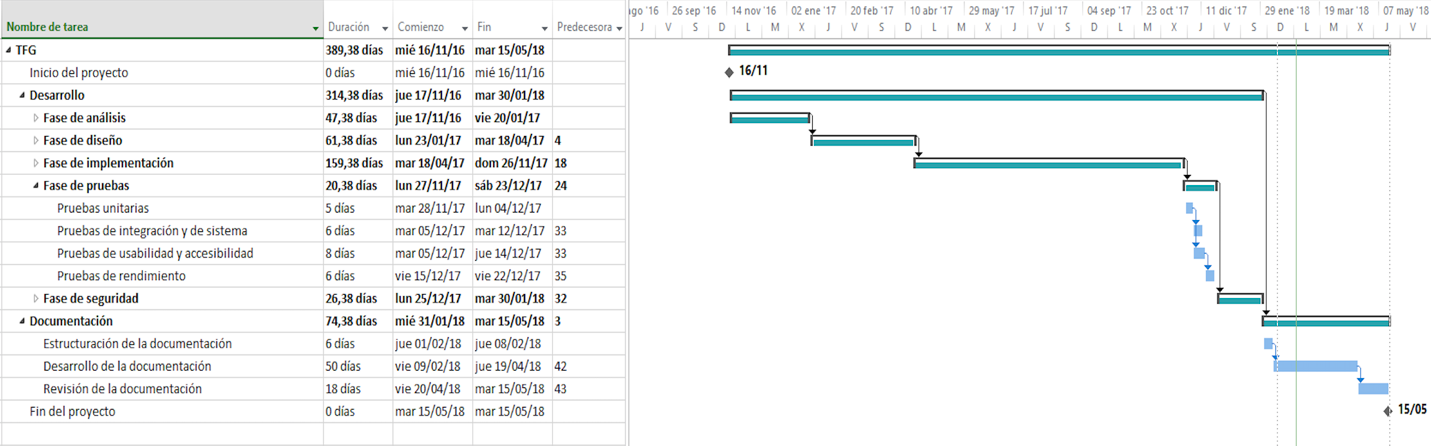


Figura 10 Diagrama Gantt - Fase de pruebas

2.3 Presupuesto

El presupuesto del presente trabajo fin de grado se divide en dos partes. La primera de ellas corresponde al coste de los elementos hardware y software del sistema. La segunda parte corresponde al coste del personal involucrado en el desarrollo del mismo. En los siguientes apartados se presentará una estimación detallada del coste de las dos partes anteriormente citadas y del coste total que conlleva el desarrollo de este trabajo fin de grado.

2.3.1 Coste del material software/hardware

El coste del material software y hardware utilizado en el desarrollo de la aplicación web se realizará en este primer apartado.

El desarrollo de la aplicación web ha sido íntegramente desarrollado con tecnologías *open source* por lo que la parte software no tiene coste alguno. Por otro lado, el IDE (entorno integrado de desarrollo) usado en el desarrollo de todo el proyecto ha sido el Visual Studio Code, un editor de código gratuito que tampoco supone ningún coste al proyecto.

Un aspecto importante a destacar es que el equipo con el que se desarrolla la aplicación web y con el que se realiza la documentación solo se utilizará durante este periodo de tiempo. Por tanto, el coste que supone utilizar el equipo durante el tiempo de desarrollo y documentación del proyecto se debe calcular a partir de su coste de amortización. Según la Agencia Tributaria y como se especifica en la tabla de coeficientes de amortización lineal, aquellos equipos electrónicos e informáticos catalogados como equipos para procesos de información tienen un gasto de amortización del 25% anual (con un máximo de 8 años). Por lo tanto, el equipo **MacBook Pro 15’’ del año 2016 con un coste total de compra de 2.255,59€ (IVA incluido), tiene un gasto de amortización asociado de:**

2.255,59€ x 25% = 563,89€ anuales.

Como se ha utilizado 1,67 años, se deduce un gasto final de:

563,89€ x 1,67 años = **939,82€** deducibles degastos de amortización.

Además del equipo, se tiene que añadir el gasto de la conexión a internet, por lo que se ha incluido en el presupuesto una tarifa de red necesaria para el desarrollo de la aplicación (descarga de librerías, actualizaciones, consulta de documentación, etc.).

Cabe destacar que, si se decide alojar la aplicación web en un hosting diferente o si se supera la capacidad gratuita del hosting actual, se debería contratar un servicio de hosting adecuado y un dominio específico para alojar la aplicación, lo cual supondría un coste adicional. Los siguientes enlaces muestran ejemplos de tarifas competitivas actuales:

* <https://raiolanetworks.es/hosting/>
* <https://sered.net/hosting-espa%C3%B1a-ssd>
* <https://www.adjenet.net/hosting>
* <https://firebase.google.com/pricing/>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***#*** | **Descripción** | **Unidad** | **Cantidad** | **Precio/Ud. (€)** | **Total** |
| *1* | MacBook Pro 15’’ 2016 Intel i7 - 16GB | Ud. | 1 | 939,82 € | 939,82 € |
| *2* | Tarifa de red | Mes | 20 | 25 € | 500 € |
| *3* | Sistema operativo (macOS High Sierra) | Ud. | 1 | 0 € | 0 € |
| *4* | Framework (Angular) | Ud. | 1 | 0 € | 0 € |
| *5* | IDE (Visual Studio Code) | Ud. | 1 | 0 € | 0 € |
|  |  |  |  | ***SUBTOTAL*** | **1.439,82 €** |

Tabla 1 Coste del material software/hardware

2.3.2 Coste del personal

Para calcular el coste del personal involucrado en el desarrollo del proyecto se hace una distinción de dos perfiles distintos. Por un lado, el perfil de analista informático será el encargado de desarrollar la fase de análisis y el diseño del proyecto, así como su inclusión en la documentación. Por otro lado, el perfil de desarrollador informático será el encargado de la fase de implementación y pruebas, así como también su inclusión en la documentación. Además, el desarrollador será el encargado de realizar los manuales de usuario e instalación.

Para calcular el precio final de este apartado se ha estimado una jornada de trabajo de 2 horas diarias durante toda la duración del proyecto (excluyendo días no laborales y festivos). El 70% del tiempo se ha destinado a la realización de las tareas correspondientes de cada fase y el 30% restante a la documentación. El precio para el personal es de 15€/hora.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Descripción** | **Cantidad (días)** | **Precio/Ud.** | **Total** |
| *1* | Analista informático | - | - | - |
| *1.1* | Fase de análisis | 45 | 30 € | 1.350 € |
| *1.2* | Fase de diseño | 53 | 30 € | 1.590 € |
| *1.3* | Documentación | 50 | 30 € | 1.500 € |
| *2* | Programador informático | - | - | - |
| *2.1* | Fase de implementación | 160 | 30 € | 4.800 € |
| *2.2* | Fase de pruebas | 30 | 30 € | 900 € |
| *2.3* | Documentación | 55 | 30 € | 1.650 € |
|  |  |  | **SUBTOTAL** | **11.790 €** |

Tabla 2 Coste del personal

2.3.3 Coste total

Por último, en este apartado se calculará el coste total del proyecto, siendo este la suma del coste de material software/hardware y el coste del personal involucrado. Nótese que además se aplica el impuesto sobre el valor añadido (I.V.A.) al subtotal calculado.

|  |  |  |  |
| --- | --- | --- | --- |
| ***#*** | **Descripción** | **Precio** | **Total** |
| *1* | Coste del material (software/hardware) | 1.439,82 € | 1.439,82 € |
| *2* | Coste del personal | 11.790 € | 11.790 € |
|  |  | ***SUBTOTAL*** | **13.229,82 €** |
|  |  | *I.V.A (21%)* | 2.778,26 € |
|  |  | **TOTAL** | **16.008,08 €** |

Tabla 3 Coste total

# DOCUMENTO 3: REQUISITOS DE USUARIO Y ANÁLISIS DE ALTERNATIVAS

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

3.1 Introducción

Este tercer documento desglosa con detalle la parte de análisis del proyecto, siendo una de las partes más importantes de cualquier proyecto de desarrollo software. Se definirán los diferentes roles que puede tener un usuario de la aplicación, así como los requisitos de usuario, un estudio más detallado de las tecnologías y herramientas a utilizar y finalmente la alternativa seleccionada.

3.2 Identificación de usuarios participantes en el sistema

La aplicación contará con dos roles de usuario bien definidos. Ambos podrán usar la misma aplicación, pero los roles son independientes. No compartirán ninguna funcionalidad, excepto la página de inicio de la aplicación y el *login*.

3.2.1 Rol de participante en el congreso

La aplicación web permitirá crear un perfil de usuario para todos los participantes del congreso anual a través de un registro en el que deberán introducir unos datos mínimos de contacto. En el perfil creado los usuarios podrán modificar sus datos personales, modificar la contraseña, adjuntar justificante de pago, añadir datos para la factura, subir artículos y descargar documentos asociados al congreso.

3.2.2 Rol de usuario administrador

El usuario administrador podrá acceder a la aplicación a través del *login*. Las credenciales (usuario y contraseña) están predefinidas, pero podrán ser modificadas dentro de la aplicación únicamente por el usuario administrador. Además, en este rol se podrá ver y modificar todos los usuarios registrados, los artículos subidos, los pagos y su aceptación, las conferencias disponibles y los autores de los artículos. Por último, el usuario administrador podrá descargar todos los documentos asociados a la conferencia y cambiar la configuración global de toda la aplicación y de la conferencia.

3.3 Requisitos de usuario

Los requisitos de usuario tienen una importancia muy alta en el desarrollo de cualquier producto software. Son declaraciones en lenguaje natural de los servicios que un usuario espera y que el sistema debe proveer. Además, especifican con detalle todas las restricciones de la aplicación.

A continuación, se detallarán los requisitos de usuario divididos en tres apartados: requisitos de usuario comunes a los dos roles de usuario existentes, requisitos de usuarios propios del rol de participante en el congreso y por último requisitos de usuario propios del rol de usuario administrador.

3.3.1 Requisitos de usuario comunes

* *Login* en la aplicación con el nombre de usuario y la contraseña. El usuario accederá a las funciones únicas de su rol y tendrá acceso solo a sus datos.
* *Logout* de la sesión de usuario. El usuario saldrá a la pantalla de *login* y cerrará su sesión, por lo que tendrá que volver a iniciar sesión si desea volver a la pantalla anterior.
* Navegación a la web organizadora del congreso. Desde la pantalla inicial el usuario podrá navegar a dicha web.
* Cambiar idioma de la aplicación. Por defecto la aplicación se encontrará en inglés, pero el usuario podrá cambiar el idioma desde cualquier pantalla y en cualquier momento.

3.3.2 Requisitos de usuario – rol de participante en el congreso

* Registro en la aplicación. Se deberán añadir los datos personales del usuario, así como una dirección email válida y una contraseña con unos requisitos de seguridad mínimos. Algunos de los campos del registro serán opcionales, pero para validar el registro se deberán aceptar las condiciones del congreso. El registro no se efectuará a menos que se rellenen los campos obligatorios del usuario. El usuario recibirá un correo informando de su registro en la aplicación.
* Modificar los datos personales del usuario, excepto la dirección email. No se podrá dejar ningún campo obligatorio del usuario vacío durante la modificación.
* Modificar la contraseña del usuario. El usuario deberá introducir su contraseña actual y su nueva contraseña con una verificación de seguridad.
* Solicitar nueva contraseña al sistema. Si el usuario no recuerda la contraseña, deberá introducir su correo electrónico y recibirá una contraseña aleatoria nueva vía email.
* Subir justificante de pago a la aplicación. El usuario podrá subir un justificante de pago almacenado de forma local y podrá modificarlo siempre que lo desee. El sistema recibirá un email indicando que el usuario ha subido un justificante de pago.
* Añadir datos de factura, todos sus campos serán obligatorios para poder almacenarlos en la aplicación. La factura asociada al usuario tendrá estos datos.
* Subir artículo a la aplicación añadiendo los datos relativos al artículo y a los autores del artículo. Esta acción enviará un correo a la dirección del congreso indicando la subida de un nuevo artículo.
* Modificación de los datos y el archivo de los artículos subidos por el usuario. Dependiendo del estado del artículo, es posible que la modificación no esté habilitada. Todos los campos obligatorios deben estar cubiertos para la modificación de cualquier artículo. Además, dependiendo del estado del artículo dichos cambios podrán implicar el envío de un email al sistema indicando los cambios realizados por el usuario. Dependiendo del estado del artículo, este podrá ser eliminado por el usuario.
* Descarga de los documentos asociados al congreso para el usuario. Los documentos descargables son: factura, certificado de ponencia y certificado de asistencia. Estos documentos serán descargables cuando se haya validado el pago por parte del administrador. Además, solo se habilitará el certificado de ponencia si el usuario ha subido algún artículo y este ha sido aceptado por el congreso.

3.3.3 Requisitos de usuario – rol de usuario administrador

* Visión global y modificación de todos los usuarios registrados en el sistema. Se podrán modificar todos los datos de los usuarios excepto el email y la contraseña (no accesible). También se podrá eliminar el usuario.
* Visión global y modificación de pagos de los usuarios registrados. Se podrá descargar el justificante de pago de cada usuario, se podrá cambiar el precio a pagar por la asistencia al congreso para cada usuario, se podrá verificar la validez del justificante de pago y se podrá descargar la factura individual de cada usuario.
* Visión global de los autores principales de los artículos subidos. Se indicará el número de artículos en los que participa cada autor y el número de posters. No se podrán hacer modificaciones en esta parte.
* Visión global y modificación de los datos de todos los artículos. Se podrá descargar el documento subido por cada usuario. También se podrá cambiar el estado de un artículo, lo que enviaría un email tanto al usuario como al sistema indicando el cambio de estado del artículo. Además, se podrá modificar todos los datos de cada artículo en cualquier momento. Por último, se podrá eliminar el artículo, por lo que el usuario no verá dicho artículo en la aplicación.
* Visión global y modificación de conferencias. Se mostrarán todas las conferencias existentes para el congreso actual, el número de artículos pertenecientes a cada conferencia y el número de posters. Se podrá modificar el título de la conferencia, se podrá añadir una nueva conferencia y se podrá eliminar una conferencia ya existente.
* Descarga de documentos relativos a la conferencia. Por un lado, se podrá descargar todos los certificados de asistencia de todos los usuarios registrados, todos los certificados de ponencia de todos los usuarios registrados y todas las facturas de todos los usuarios registrados, todos ellos en formato PDF. Por otro lado, se podrá descargar en formato Excel la lista de todos los usuarios registrados, la lista de todos los artículos subidos y la lista de todos los autores participantes. Por último, se podrá generar un PDF con etiquetas identificativas para todos los usuarios asistentes al congreso. Estas etiquetas podrán ser modificadas en color y tamaño a elección del usuario administrador y podrán llevar un texto en lugar del nombre de los usuarios participantes.
* Visión general y modificación de la configuración de la aplicación. Se podrá consultar y modificar todos los parámetros de configuración como el usuario y la contraseña de administrador, el año de la conferencia, las fechas de inicio y clausura, el lugar donde será celebrado el congreso, la dirección web oficial del congreso y otros datos. Además, será posible modificar la configuración de correo electrónico y/o cerrar o abrir el servicio de correos. Por último, se podrá reiniciar la base de datos de la aplicación y cerrar o abrir tanto el acceso general como el periodo de subida de artículos a la aplicación.

3.4 Análisis de alternativas

En este apartado se hará un estudio de las diferentes herramientas y lenguajes más usados hasta hoy en la programación web. En primer lugar, se detallará una breve descripción de los lenguajes de programación y sus correspondientes *frameworks* que forman parte del lado del servidor (*backend*). En segundo lugar, se hará lo correspondiente con los lenguajes de programación y los *frameworks* asociados en la parte del cliente (*frontend*). Por último, se mostrará un resumen y descripción de las bases de datos más usadas.

3.4.1 Tecnologías *backend*

La parte del lado del servidor o tecnología *backend* es la capa intermedia entre la base de datos y el código que el navegador web va a ejecutar. Estos lenguajes de programación son ejecutados desde el servidor donde está alojada la aplicación. Se encargan de tratar con los datos de la base de datos y de enviarlos al *front* para que estos sean mostrados en el navegador.

**3.4.1.1 PHP**

A día de hoy es el lenguaje más usado para el desarrollo web. Se trata de un lenguaje de código abierto con más de 20 años de antigüedad, basado en los lenguajes C# y PERL. Cuenta con una gran comunidad de desarrolladores y usuarios desde sus inicios, además de numerosas aplicaciones web de éxito en las que se ha utilizado este lenguaje (Yahoo, Wikipedia, Facebook, Wordpress o Flickr).

Entre las ventajas de este lenguaje está la continua demanda por parte de las empresas y su compatibilidad con los sistemas operativos y plataformas del mercado. Además, cuenta con varios *frameworks* de desarrollo como Laravel, Symfony2, Yii, etc.



Figura 11 Logo PHP

**3.4.1.2 Ruby**

Lanzado al mercado en 1995, Ruby es un lenguaje de programación orientado a objetos. Ruby on Rails es el *framework* de desarrollo que se utiliza para construir aplicaciones web bajo el lenguaje Ruby. Este *framework* es de código abierto y, como PHP, cuenta con muchos usuarios a día de hoy por lo que la comunidad también es extensa. Twitter, Github, LinkedIn o SlideShare son algunas de las aplicaciones desarrolladas en Ruby on Rails. Entre las ventajas de utilizar este *framework* para el desarrollo web está la rapidez y facilidad que aporta el entorno de trabajo para construir la aplicación, así como también la sencillez para incluir paquetes de librerías.

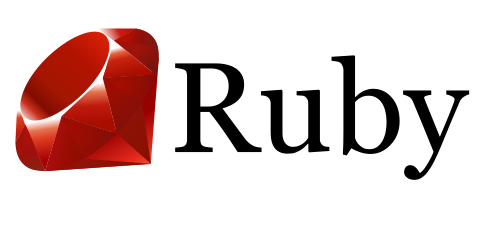


Figura 12 Logo Ruby

**3.4.1.3 Python**

Se trata de un lenguaje interpretado y multiparadigma creado a finales de los 80. Python se caracteriza por ser un lenguaje con tipificación dinámica y multiplataforma. La comunidad de desarrolladores también es muy alta. Es de código abierto y su filosofía es favorecer el desarrollo de código legible.

No es el lenguaje más utilizado en la programación web, pero Python cuenta un potente *framework* web de alto nivel con una amplia documentación llamado Django. Instagram, Pinterest o The New York Times son algunas de las aplicaciones web que se han desarrollado con este lenguaje.

****

Figura 13 Logo Python

**3.4.1.4 ASP.NET**

ASP.NET es un entorno para aplicaciones web propiedad de Microsoft. Creado en 2002, es el sucesor de ASP. En este entorno se permite usar cualquier lenguaje admitido en el .NET *framework*, siendo el más común C# o C Sharp. Al ser propiedad de Microsoft, no es de código abierto por lo que se requiere de una licencia para su uso. Entre las ventajas de su uso está la compatibilidad con módulos de Microsoft como SQL Server, o el lenguaje Visual Basic. Además, este *framework* también ofrece tecnologías *frontend*.



Figura 14 Logo ASP.NET

**3.4.1.5 Java EE**

Java Enterprise Edition, Java EE o J2EE es una plataforma de programación destinada a desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java. En ella se permiten utilizar arquitecturas de N capas distribuidas y se apoya en componentes modulares ejecutados sobre un servidor de aplicaciones. Los beneficios de usar esta tecnología son: código abierto en la gran mayoría de herramientas que la usan, miles de librerías para extender la plataforma y simplificar el desarrollo.

Algunas de los *frameworks* más usados y destinados al desarrollo web que se integran con los estándares de la plataforma Java EE son: JavaServer Faces framework, Spring framework o Struts framework.



Figura 15 Logo Java EE

**3.4.1.6 Node.js**

Node.js es un entorno en tiempo de ejecución multiplataforma y de código abierto creado en 2009 orientado en el lado del servidor y que utiliza ECMAScript (JavaScript) como lenguaje. Asíncrono y con una arquitectura orientada a eventos, está basado en el motor V8 de Google. Creado específicamente para ser útil en la creación de programas de red altamente escalables como los servidores web. Una de las ventajas de Node.js es que puede ser combinado con una base de datos y mediante elementos JSON permite desarrollar un entorno JavaScript unificado. Adaptando los patrones para desarrollo del lado del servidor facilitando la reutilización de código del mismo modelo de interfaz entre el lado del cliente y el lado del servidor. Se trata del más moderno comentado hasta ahora y el que más futuro tiene de cara a las tecnologías web. A pesar de su juventud, la comunidad de desarrolladores es muy activa.



Figura 16 Logo Node.js

3.4.2 Tecnologías *frontend*

Las tecnologías *frontend* son la capa más alta de una aplicación web. Son las encargadas de recibir los datos finales desde las capas más bajas y construir mediante diversos lenguajes una interfaz que el navegador mostrará al usuario. Es importante destacar que las tecnologías de *frontend* son muy importantes ya que el usuario final interactúa directamente con esta capa, por lo que la interfaz debe ser moderna, usable y accesible. A continuación, se hará un resumen detallado de las tecnologías que más se utilizan hoy en día.

3.4.2.1 HTML5

HTML (HyperText Markup Language) es un estándar que define una estructura básica y un código por el que se define el contenido de una página web. El principal objetivo de este lenguaje es dar formato a los diversos contenidos de una página web para que un navegador los interprete. HTML5 es la última versión de este lenguaje y desde el 2014 es pública.



Figura 17 Logo HTML5

3.4.2.2 XML

XML (eXtensive Markup Language) es un metalenguaje que permite definir lenguajes de marcas. Fue desarrollado por el W3C (World Wide Web Consortium) y sirve para organizar contenidos y describir la información de una página web. Aunque sus características puedan ser similares a las del HTML, en general el lenguaje XML suele funcionar como un complemento del lenguaje HTML.

****

Figura 18 Logo XML

3.4.2.3 JavaScript

JavaScript o JS, es un lenguaje de programación interpretado. Principalmente se utiliza en el lado del cliente implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque también puede ser usado en el lado del servidor, como es el caso de Node.js. Mayormente influenciado por Java, también tiene influencias del C, Python o Scheme. Se ha convertido en uno de los lenguajes más populares de la programación web y con él se pretende mejorar la experiencia del usuario.



Figura 19 Logo JavaScript

3.4.2.4 jQuery

jQuery es una librería multiplataforma de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM (Doment Object Model), manejar eventos, desarrollar animaciones y en general contenido dinámico a la aplicación web. Desarrollada en 2006 y de código abierto, se trata de la librería JavaScript más utilizada.



Figura 20 Logo jQuery

3.4.2.5 TypeScript

TypeScript es un lenguaje de programación de código abierto desarrollado y mantenido por Microsoft sacado al mercado en 2012. Se trata de un superconjunto de JavaScript, que básicamente añade tipificación estática y objetos basados en clases, lo que le hace un lenguaje orientado a objetos. Cualquier código JavaScript funciona en TypeScript. A través de un compilador, el código de TypeScript es traducido a JavaScript para que los navegadores puedan ejecutar dicho código.



Figura 21 Logo TypeScript

3.4.2.6 Angular

Angular es un *framework* que ha evolucionado mucho en los últimos años. Es propiedad de Google y es de código abierto. Su primera versión llamada AngularJS fue lanzada en 2010. El lenguaje en el que se basa su primera versión es JavaScript. En el año 2016 se lanzó su segunda versión llamada Angular 2. Actualmente se ha lanzado la versión Angular 6 aunque comúnmente se le denomina Angular ya que desde su versión dos las actualizaciones no cambiaron la arquitectura del *framework* por lo que, comúnmente, se llama Angular a cualquier versión posterior. Cada cierto tiempo, los desarrolladores lanzan una nueva versión del *framework* con nuevas características y mejoras. Desde la segunda versión el lenguaje en el que se basa es TypeScript. Aún así, Angular sirve al navegador el código TypeScript de la aplicación traducido a JavaScript para que los navegadores puedan ejecutar dicho código. Angular sigue el patrón arquitectónico MVC ya que se basa en la creación de componentes y servicios. Cada componente tiene un archivo HTML, un archivo Typescript y un archivo CSS o SCSS. Esto permite organizar el código y es Angular el que se encarga de juntar todos los ficheros en la compilación.



Figura 22 Logo Angular

3.4.2.7 CSS3

CSS (Cascading StyleSheets) es un lenguaje utilizado para definir los estilos de una aplicación web. Este lenguaje se suele combinar con HTML para definir la interfaz del usuario. Actualmente se utiliza su última versión CSS3. Se basa en una modularización de ficheros que contienen nombres de clases incluidos en el DOM (Definition Object Model). Las clases son aplicadas siguiendo un sistema de puntuación. La clase con mayor valor es la que hace efecto en el DOM, invalidando las clases que afectan al mismo elemento en el DOM con menor valor.



Figura 23 Logo CSS3

3.4.2.8 Sass

Sass es un metalenguaje de CSS. Es un lenguaje de script que es traducido a CSS. La sintaxis más reciente, SCSS, usa el formato de bloques como CSS. Se trata de un lenguaje precompilado de hojas de estilo muy útil en la actualidad para la maquetación de aplicaciones web. A diferencia del lenguaje CSS, Sass permite la definición de variables en sus ficheros, la creación de *mixins*, anidamiento de clases y herencia de selectores entre otras muchas cosas. Estos ficheros son precompilados a CSS para servirlos posteriormente al navegador como clases CSS comunes.



Figura 24 Logo Sass

3.4.2.9 Bootstrap

Bootstrap es un framework de código abierto lanzado en el año 2011. Se trata de una de las mejores herramientas de maquetado y diseño para aplicaciones web que existen en la actualidad. Bootstrap proporciona un conjunto de plantillas, *grids* y otros elementos basados en HTML y CSS, así como de extensiones JavaScript adicionales.



Figura 25 Logo Bootstrap

3.4.3 Base de datos

La base de datos es la capa más baja de una aplicación web. En ella se almacenan todos los datos relevantes al sistema. En el caso de la aplicación planteada en el presente trabajo fin de grado, se guardarán todos los datos de todos los usuarios registrados, incluyendo los justificantes de pago y los artículos de cada usuario. Además, se almacenarán las conferencias y la configuración de la aplicación.

3.4.3.1 MySQL

Es un sistema de gestión de bases de datos relacional y está considerada como la base de datos de código abierto más popular del mundo. Es soportada por los lenguajes de programación más importantes y la comunidad de desarrolladores es muy amplia. Wikipedia, Google, Facebook, Twitter, Flickr o YouTube son algunos ejemplos de sitios web que la utilizan.



Figura 26 Logo MySQL

3.4.3.2 PostgreSQL

Es un sistema de gestión de bases de datos relacional orientado a objetos y de código libre. Cuenta con alta concurrencia, amplia variedad de tipos nativos, amplia seguridad, etc. Al igual que con MySQL, PostgreSQL da soporte a la mayoría de los lenguajes de programación.



Figura 27 Logo PostgreSQL

3.4.3.3 MongoDB

Es un sistema de base de datos NoSQL orientado a documentos y de código abierto. Lanzado en 2009, los datos son guardados en estructuras de datos de documentos similares a JSON en lugar de tablas como en los sistemas relacionales tradicionales. MongoDB utiliza una especificación llamada BSON, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida. Al igual que las relacionales comentadas anteriormente, MongoDB tiene *drivers* oficiales para la mayoría de los lenguajes de programación más extendidos a día de hoy.



Figura 28 Logo MongoDB

3.4.3.4 Firebase Cloud Firestore

Es un servicio derivado de Google Cloud Platform, adaptado a la plataforma Firebase adquirida por Google en 2014. Se trata de una base de datos NoSQL que organiza sus datos en forma de documentos agrupados en colecciones. En ellos se pueden incluir campos de diversos tipos (cadenas de texto, números, puntos geográficos, referencias a la propia base de datos, booleanos, etc.) y otras subcolecciones. Junto con el resto de las herramientas de Firebase y de Google Cloud Platform, las posibilidades de esta base de datos en una aplicación web son inmensas.



Figura 29 Logo Firebase

3.5. Descripción de la alternativa seleccionada

Después de analizar todas las alternativas para cada una de las capas de la aplicación, se escogerán aquellas tecnologías que más se ajustan al desarrollo de este proyecto. Cabe destacar que el presente trabajo fin de grado se titula “Mejora del Rendimiento y Ampliación de una Web para la gestión integral de Congresos Científicos”, por lo que una de las premisas es la de modernizar toda la tecnología ya existente. Además, la aplicación tiene un alcance exclusivo para la web, por lo que deberán ser seleccionadas aquellas tecnologías que mejor se adapten a una aplicación web.

Las tecnologías web avanzan con una rapidez mucho mayor que cualquier otra tecnología, por eso es recomendable apostar por tecnologías modernas que tienen dentro de su *roadmap* una mejora continua y un soporte técnico de un gran número de profesionales.

A continuación, se detallarán las razones de las tecnologías seleccionadas para cada una de las capas: *backend*, *frontend* y base de datos.

3.5.1 Tecnologías *frontend*

Tras analizar todas las posibilidades tratadas en el apartado anterior se ha decidido usar el moderno *framework* Angular, en su versión 5. Las razones se detallan a continuación:

* Este *framework* sigue el patrón MVC proporcionando una estructura organizada a la aplicación web. El comienzo del proyecto es sencillo ya que Angular crea un esqueleto mínimo que puede ser ejecutado con un simple comando ‘*ng serve -o*’.
* Angular puede usar diferentes gestores de paquetes. Destacan npm y yarn, ambos con miles de paquetes disponibles para cualquier aplicación desarrollada con Angular. Se basa en software libre por lo que cualquier persona puede subir su propia librería y hacerla pública para el resto de los usuarios. El uso de estos gestores simplifica el desarrollo de una aplicación ya que muchas de las librerías disponibles fueron desarrolladas para añadirlas a aplicaciones web de la misma índole.
* Angular se basa principalmente en la creación y reutilización de componentes. Cada componente está compuesto por cuatro ficheros: la vista, la lógica, los estilos y los test. La vista es un fichero HTML. La lógica es un fichero TypeScript. Los estilos es un fichero CSS o SCSS. Los test es un fichero TypeScript con una sintaxis de validación. Todos estos ficheros forman parte del componente y éste puede ser usado en la aplicación añadiendo el selector que se encuentra en la lógica a cualquier otro fichero de la vista de cualquier otro componente. Esto quiere decir que los componentes pueden usarse unos dentro de otros, lo que simplifica la programación y la hace más limpia, evitando duplicidad de código.
* Otra de las razones es que Angular es mantenido por Google por lo que tiene un soporte continuo de los mejores profesionales del sector, siendo actualizado día a día con nuevas características y versiones del *framework.* Aún así, es de código abierto y por ello la comunidad actual de Angular es muy grande y tiende a ser cada vez mayor, los programadores web son atraídos por la agilidad de mejora, el rendimiento y la sencillez para crear aplicaciones web potentes y modernas.
* Angular a partir de su versión cinco incluye su propio repositorio de componentes ya definidos que son totalmente personalizables: Angular Material. Estos componentes están optimizados para que ofrezcan el mejor rendimiento a las aplicaciones web.
* Angular Universal, propiedad de Angular permite ejecutar la primera vista de la aplicación en el lado del servidor para una adaptación casi instantáneo y para la mejora del SEO en la aplicación.
* Angular CLI, herramienta de línea de comandos que permite realizar tareas tediosas con simples comandos, ayudan significativamente el desarrollo de cualquier aplicación.
* Angular posee decenas de *plugins* para los IDEs de desarrollo web más comunes. Estos *plugins* facilitan enormemente el desarrollo. El IDE, Visual Studio Code, tiene numerosas ventajas como la sugerencia de código inteligente y detección de errores además de múltiples *plugins*.
* Angular permite fácilmente la integración con otras tecnologías o *frameworks* web como jQuery, Bootstrap, HTML5, CSS3, JavaScript o Sass que también serán utilizadas en el lado del cliente de la aplicación.
* Angular e Ionic, *framework* para la creación de aplicaciones móviles, son como uña y carne para pasar aplicaciones web a su versión móvil. Con tan solo unos comandos de instalación y de ejecución, cualquier aplicación escrita en Angular puede pasar a ser una aplicación móvil de Ionic.

3.5.2 Tecnologías *backend*

En el lado del servidor todas las tecnologías comentadas podrían ser usadas ya que todas ellas cumplirían con las expectativas previstas. Sin embargo, se ha escogido Node.js ya que es la que mejor se adapta con Angular. Las razones por las que se ha elegido son:

* Tanto Node.js como Angular comparten indirectamente el mismo lenguaje de programación. Aunque en Angular se use TypeScript, el código TypeScript es similar al código JavaScript de Node.js y cualquier fragmento de código JavaScript funcionaría también en TypeScript. Esto es útil si se desea replicar código del lado del servidor en el lado del cliente.
* Tanto Node.js como Angular poseen la misma estructura de intercambio de datos: JSON. Cualquier comunicación entre el cliente y el servidor o viceversa es directa a través de ficheros JSON.
* Al igual que Angular, Node.js puede usar npm o yarn como gestor de paquetes.
* Node.js posee una gran comunidad *open source* a sus espaldas que evoluciona ágilmente. A pesar de su corta vida, el futuro de Node.js es muy favorable gracias a ello.
* El navegador más usado a día de hoy, Google Chrome, usa el mismo entorno de ejecución escrito en JavaScript que Node.js. Se trata de V8. Esto hace que las aplicaciones de Node.js estén optimizadas cuando se usa el navegador más usado en el mundo.
* Node.js funciona en un único hilo de ejecución, con entradas y salidas asíncronas que pueden ejecutarse concurrentemente hasta en cientos de miles sin acarrear costos asociados al cambio de contexto. Por ello escala increíblemente bien cuando una aplicación supera inesperadamente el número de visitas o el número de peticiones.

3.5.3 Base de datos

Una vez escogidas las tecnologías *backend* y *frontend* de la aplicación, hay que escoger el sistema de almacenamiento que mejor se adapte a las tecnologías elegidas. En este caso, y ya que se ha apostado por usar las tecnologías más modernas del mercado, una de las tendencias actuales para las bases de datos es Firebase, plataforma integrada en Google Cloud Platform que cuenta con muchas herramientas útiles para el desarrollo web, entre las que se incluyen las bases de datos. Arriesgando un poco, se ha elegido Cloud Firestore como base de datos NoSQL o no relacional en su versión BETA. Las razones son las siguientes:

* Cloud Firestore es propiedad de Google. Al formar parte de Google, el soporte y la documentación es muy extensa y cuenta con numerosos tutoriales.
* Como parte de Firebase, la integración con otras herramientas de Firebase como DebugView, Audiences, Functions, AdMob, Predictions o muchas otras, enriquece enormemente las posibilidades de la aplicación.
* Storage, también de Firebase, al igual que Cloud Firestore almacena en la nube todos los archivos de la aplicación por carpetas.
* En cualquier momento, en cualquier lugar y con cualquier dispositivo con una conexión a internet es posible acceder, modificar o eliminar cualquier dato o archivo de la aplicación.
* Cloud Firestore almacena los datos en colecciones de documentos, al estilo JSON. Las colecciones pueden ser almacenadas en otras colecciones, por lo que la escalabilidad es muy buena.
* Firebase ofrece unos parámetros de configuración para aplicaciones web, aplicaciones Android y aplicaciones iOS a la misma base de datos. Si la aplicación se desarrolla en un futuro para móvil, esto agiliza enormemente el tiempo de integración con la base de datos, ya que la implementación es inmediata con estos parámetros de configuración.
* Cloud Firestore usa la sincronización de datos para actualizar los datos de cualquier dispositivo conectado, usando consultas de recuperación únicas de manera eficiente.
* Cloud Firestore cuenta con asistencia sin conexión. Los datos se almacenan en la caché de la aplicación de forma activa, aunque no haya conexión. Cuando se recupera la conexión, Cloud Firestore sincroniza todos los cambios cacheados.

3.5.4 Sistema operativo

Las tecnologías escogidas de todas las capas son multiplataforma, por lo que cualquier sistema operativo puede ejecutar el código de la aplicación. El usuario del sistema podrá escoger el sistema operativo que más le guste para la ejecución de la aplicación o el desarrollo en local.

# DOCUMENTO 4: ANÁLISIS DE REQUISITOS DEL SISTEMA

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

4.1 Introducción

En este documento se hará una identificación de los subsistemas de análisis que descompondrán el sistema con el objetivo de facilitar el análisis global de éste. Una vez identificados, se procederá a la obtención de requisitos de sistema mediante el desarrollo de casos de uso.

Los casos de uso describen las funciones que la aplicación debe ejecutar cuando el usuario interactúa con el sistema. Normalmente se desglosa en una secuencia de interacciones entre el sistema y el usuario.

4.2 Subsistemas de análisis y casos de uso

Los subsistemas de la aplicación web que conformarán nuestro sistema principal son los siguientes:

1. Acceso y registro de usuarios.
2. Gestión de usuarios asistentes al congreso.
3. Gestión de usuario administrador.

A continuación, se describirán los casos de uso de cada subsistema. Para cada subsistema se incluirá un diagrama para dar una visión global del subsistema.

4.2.1 Acceso y registro de usuarios

En este apartado se desarrollarán los casos de uso para el subsistema “Acceso y registro de usuarios”, que engloba aquellas funcionalidades relacionadas con el acceso a la aplicación web (registro y *login*). En la siguiente figura se detallan los principales casos de uso de este subsistema:

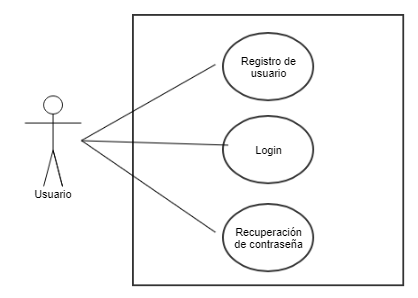


Figura 30 Subsistema Acceso y registro de usuarios

4.2.1.1 Registro de usuario

* Actores: usuarios asistentes al congreso.
* Precondiciones: que el usuario no esté registrado en la aplicación.
* Descripción: el usuario se podrá registrar en la aplicación creando su propio perfil de usuario.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla principal de la aplicación web.

2. El usuario accede a la página de registro.

3. El usuario rellena el formulario de registro.

4. El usuario envía el formulario rellenado.

5. El sistema valida la información y registra al usuario en la aplicación.

6. El sistema notifica al usuario vía mail de que se ha registrado correctamente. El sistema también notificará vía mail al usuario administrador.

* Excepciones:

4. El usuario navega a otro apartado o página de la aplicación antes de clicar en el botón de envío del formulario. El registro no se producirá y el usuario no se registrará en el sistema.

5. El sistema valida la aplicación y detecta que el correo electrónico pertenece a otro usuario ya registrado en la aplicación. El sistema notifica al usuario que el usuario ya existe.

4.2.1.2 *Login*

* Actores: usuarios asistentes al congreso o usuario administrador.
* Precondiciones: el usuario debe estar registrado en la plataforma.
* Descripción: el usuario iniciará sesión en la aplicación web.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla principal de la aplicación web.

2. El usuario accede a la página de *login*.

3. El usuario rellena el nombre de usuario (email) y contraseña.

4. El usuario clica en el botón de inicio de sesión o pulsa la tecla *enter*.

5. El sistema valida los datos e inicia la sesión del usuario.

* Excepciones:  
  4. El usuario navega a otro apartado o página de la aplicación antes de clicar en el botón de inicio de sesión o pulsar la tecla *enter*. Los datos introducidos no se validarán y el usuario no iniciará sesión.

5. El sistema valida los datos de inicio de sesión y éstos son incorrectos, el sistema notificará al usuario que los datos no son correctos.

4.2.1.3 Recuperación de contraseña

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma.
* Descripción: el usuario establecerá una nueva contraseña.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla principal de la aplicación web.

2. El usuario accede a la página de *login*.

3. El usuario accede a la pantalla de recuperación de contraseña.

4. El usuario introduce su dirección de correo electrónico.

5. El usuario clica en el botón de recuperar.

6. El sistema valida la dirección y envía un email al correo electrónico previamente introducido por el usuario con una nueva contraseña generada aleatoriamente por el sistema. El sistema también actualizará la base de datos con esta nueva contraseña.

* Excepciones:  
  6. El sistema valida el correo electrónico introducido por el usuario y éste no es válido. El sistema indica al usuario que el usuario no existe.

4.2.2 Gestión de usuarios asistentes al congreso

En este apartado se analizarán los casos de uso del subsistema “Gestión de usuarios asistentes al congreso”, el cual engloba todas las funcionalidades relativas a los usuarios registrados en el sistema. En la siguiente figura se pueden ver los casos de uso principales de este subsistema:

Figura 31 Subsistema Gestión de usuarios asistentes al congreso

4.2.2.1 Modificación de la información de usuario

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en el sistema y haber iniciado sesión.
* Descripción: el usuario podrá cambiar toda la información de su cuenta a excepción del email.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla perfil.

2. El usuario clica en el botón “Cambiar datos”.

3. El usuario modifica aquellos campos de su cuenta que desee cambiar.

4. El usuario guarda los cambios clicando el botón “Guardar cambios”.

5. El sistema valida la información y actualiza los datos.

6. El sistema notifica al usuario del éxito de la operación.

* Excepciones:  
  4. El usuario accede a otro apartado o página de la aplicación antes de clicar el botón “Guardar cambios”. Los nuevos datos introducidos no se guardarán.

4.2.2.2 Modificación de la contraseña de usuario

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma y haber iniciado sesión.
* Descripción: el usuario podrá modificar su contraseña.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla perfil.

2. El usuario clica en el botón “Cambiar contraseña”.

3. El sistema abre una ventana modal con tres campos: contraseña actual, contraseña nueva y confirmar contraseña nueva.

4. El usuario introduce los datos en la ventana modal.

5. El usuario clica en el botón “Guardar cambios”.

6. El sistema guarda la nueva contraseña definida por el usuario y notifica a éste del éxito de la operación.

* Excepciones:  
  5. El usuario accede a otro apartado o página de la aplicación antes de clicar el botón “Guardar cambios”. La contraseña no se cambiará y se perderán los datos introducidos.

4.2.2.3 Subir justificante de pago

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma y haber iniciado sesión.
* Descripción: el usuario podrá subir un justificante de pago.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de pagos y facturas.

2. El usuario clica en el botón “Seleccionar archivo” y escoge un archivo almacenado en su dispositivo.

3. El sistema almacena el archivo seleccionado y notifica al usuario el éxito de la operación.

4. El sistema envía un correo electrónico al usuario administrador indicando que el usuario ha subido un justificante de pago.

4.2.2.4 Eliminar justificante de pago

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión y haber subido previamente un justificante de pago.
* Descripción: el usuario podrá eliminar el justificante de pago subido previamente.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de pagos y facturas.

2. El usuario clica en el botón “x” de la sección “Pagos”.

3. El sistema elimina el justificante de pago y notifica al usuario el éxito de la operación.

4.2.2.5 Visualización de justificante de pago

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión y haber subido previamente un justificante de pago.
* Descripción: el usuario podrá visualizar el justificante de pago subido previamente.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de pagos y facturas.

2. El usuario clica en el enlace del nombre del archivo subido.

3. El sistema abre una pestaña nueva en el navegador con el justificante de pago subido.

4.2.2.6 Añadir datos de factura

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma y haber iniciado sesión.
* Descripción: el usuario podrá añadir datos para la generación de su factura.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de pagos y facturas.

2. El usuario rellena los datos del formulario de la sección “Facturas”.

3. El usuario clica en el botón “Enviar”.

4. El sistema guarda los datos introducidos y notifica al usuario el éxito de la operación.

* Excepciones:  
  3. El usuario accede a otro apartado o página de la aplicación antes de clicar el botón “Enviar”. Los datos no se almacenarán y se perderán los datos introducidos.

4.2.2.7 Subir artículo

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma y haber iniciado sesión.
* Descripción: el usuario podrá subir un artículo a la plataforma.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de artículos.

2. El usuario rellena los datos del formulario de la sección “Nuevo artículo”. Incluyendo el archivo a subir y los autores del artículo.

3. El usuario clica en el botón “Enviar”.

4. El sistema guarda los datos introducidos, el archivo subido y notifica al usuario el éxito de la operación.

5. El sistema muestra en una tabla dentro de la sección “Mis artículos” el artículo subido.

6. El sistema envía un correo electrónico al usuario administrador indicando que el usuario ha subido un nuevo artículo.

* Excepciones:  
  3. El usuario accede a otro apartado o página de la aplicación antes de clicar el botón “Enviar”. Los datos no se almacenarán y se perderán los datos introducidos.

4.2.2.8 Visualizar artículo

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión y haber subido un artículo previamente.
* Descripción: el usuario podrá visualizar un artículo subido previamente.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de artículos.

2. En la sección “Mis artículos” se encuentra una tabla con el artículo a visualizar. El usuario clica en el icono de la columna “Archivo” de la tabla.

3. El sistema abre en una pestaña nueva del navegador el artículo seleccionado.

4.2.2.9 Modificación de artículo

Correo al sistema si el estado es Major /Minor

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión y haber subido un artículo previamente. Además, el estado del artículo debe ser “subido” o “mayor/menor”.
* Descripción: el usuario podrá modificar un artículo subido previamente.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de artículos.

2. En la sección “Mis artículos” se encuentra una tabla con el artículo a modificar. El usuario clica en el icono izquierdo (editar) de la columna “Acciones” de la tabla. El artículo está en estado “subido”.

3. El sistema abre una ventana modal con los datos del artículo. Todos los datos son modificables.

4. El usuario modifica los datos que desea del artículo.

5. El usuario clica en el botón “Guardar cambios”.

6. El sistema actualiza los datos modificados en la base de datos, cierra la ventana modal y notifica al usuario el éxito de la operación.

* Flujo alternativo:

2. En la sección “Mis artículos” se encuentra una tabla con el artículo a modificar. El usuario clica en el icono izquierdo (editar) de la columna “Acciones” de la tabla. El artículo está en estado “mayor/menor”.

3. El sistema abre una ventana modal con todos los datos del artículo modificables a excepción del “Autor principal”.

4. El usuario modifica los datos que desea del artículo.

5. El usuario clica en el botón “Guardar cambios”.

6. El sistema actualiza los datos modificados en la base de datos, cierra la ventana modal y notifica al usuario el éxito de la operación.

7. El sistema envía un correo electrónico al usuario administrador indicando que el usuario ha modificado los datos del artículo en estado “mayor/menor”.

* Excepciones:  
  5. El usuario cierra la ventana modal antes de clicar el botón “Guardar cambios”. Los datos no se almacenarán y se perderán los datos introducidos.

4.2.2.10 Eliminar artículo

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión, haber subido un artículo previamente y que el estado de éste sea “subido”.
* Descripción: el usuario podrá subir un artículo a la plataforma.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de artículos.

2. En la sección “Mis artículos” se encuentra una tabla con el artículo a eliminar. El usuario clica en el icono derecho (borrar) de la columna “Acciones” de la tabla.

3. El sistema abre una ventana modal con dos botones de acción.

4. El usuario clica en el botón “Sí”.

5. El sistema cierra el modal, elimina el artículo de la base de datos y actualiza la tabla de la sección “Mis artículos”. Además, notifica al usuario del éxito de la operación.

* Excepciones:  
  4. El usuario clica en el botón “Cancelar” o cierra la ventana modal. El artículo no se eliminará.

4.2.2.11 Descarga de factura

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión, haber subido un justificante de pago y que el usuario administrador haya validado el pago.
* Descripción: el usuario podrá descargar la factura que acredita el pago realizado previamente. Para evitar fraudes fiscales, la descarga solo podrá realizarse una única vez.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de documentos.

2. El usuario clica en el botón “Descargar” de la sección “Factura”.

3. El sistema inicia la descarga del archivo en formato PDF.

4.2.2.12 Descarga de certificado de asistencia

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión, haber subido un justificante de pago y que el usuario administrador haya validado el pago.
* Descripción: el usuario podrá descargar el certificado de asistencia al congreso. Para evitar fraudes fiscales, la descarga solo podrá realizarse una única vez.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de documentos.

2. El usuario clica en el botón “Descargar” de la sección “Certificado de asistencia”.

3. El sistema inicia la descarga del archivo en formato PDF.

4.2.2.13 Descarga de certificado de ponencia

* Actores: usuarios asistentes al congreso.
* Precondiciones: el usuario debe estar registrado en la plataforma, haber iniciado sesión, haber subido un justificante de pago, que el usuario administrador haya validado el pago, haber subido al menos un artículo que se encuentre en estado “aceptado”.
* Descripción: el usuario podrá descargar el certificado de ponencia al congreso. Para evitar fraudes fiscales, la descarga solo podrá realizarse una única vez.
* Flujo normal del caso de uso:

1. El usuario accede a la pantalla de documentos.

2. El usuario clica en el botón “Descargar” de la sección “Certificado de ponencia”.

3. El sistema inicia la descarga del archivo en formato PDF. Se añadirá una hoja al archivo por cada artículo en estado “aceptado”.

4.2.3 Gestión de usuario administrador

En este apartado se detallarán los casos de uso para el subsistema “Gestión de usuario administrador”, que engloba todas las funcionalidades relacionadas con la gestión del congreso y que solo es accesible por el usuario administrador. En la siguiente figura se pueden ver los diferentes casos de uso de este apartado:

Figura 32 Gestión de usuario administrador

4.2.3.1 Visualización tabla de usuarios

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar en forma de tabla todos los usuarios registrados.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de usuarios.

2. El sistema muestra una tabla resumen con el listado de usuarios.

4.2.3.2 Modificación usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá modificar los datos de cualquier usuario registrado.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de usuarios.

2. El usuario administrador clica en el icono izquierdo de la columna “acciones” situado en la fila del usuario que se va a modificar.

3. El sistema abre una ventana modal con los datos del usuario seleccionado deshabilitados.

4. El usuario administrador clica en el botón “Editar datos”.

5. El sistema habilita los campos con los datos del usuario.

6. El usuario administrador modifica los datos del usuario y clica en el botón “Guardar cambios”.

7. El sistema actualiza los datos modificados en la base de datos, cierra el modal y notifica al usuario administrador el éxito de la operación.

* Excepciones:  
  6. El usuario administrador cierra la ventana modal. Los datos no se cambiarán y se perderán los cambios realizados.

4.2.3.3 Eliminar usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá eliminar a cualquier usuario registrado de la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de usuarios.

2. El usuario administrador clica en el icono derecho de la columna “acciones” situado en la fila del usuario que se va a eliminar.

3. El sistema abre una ventana modal con dos acciones.

4. El usuario administrador clica en el botón “Sí”.

5. El sistema elimina al usuario de la base de datos, actualiza la tabla y notifica al usuario administrador el éxito de la operación.

* Excepciones:  
  4. El usuario administrador cierra la ventana modal. El usuario no se elimina.

4.2.3.4 Visualización tabla de pagos

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar el estado de pago de cualquier usuario registrado.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de pagos.

2. El sistema muestra una tabla resumen de pagos con el listado de usuarios.

4.2.3.5 Visualización justificante de pago de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y el usuario seleccionado debe tener justificante de pago subido.
* Descripción: el usuario administrador podrá visualizar el justificante de pago de cualquier usuario registrado de la aplicación que haya subido el archivo.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de pagos.

2. El usuario administrador clica en el icono de la columna “archivo de pago” situado en la fila del usuario elegido.

3. El sistema abre una pestaña nueva con el archivo.

4.2.3.6 Modificación cuota de pago de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá modificar la cuota de pago de cualquier usuario registrado de la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de pagos.

2. El usuario administrador modifica el campo de la tabla de la columna “a pagar” situado en la fila del usuario elegido.

3. El sistema actualiza la cuota del usuario en la base de datos y notifica al usuario administrador el éxito de la operación.

4.2.3.7 Modificación validación de pago de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y el usuario seleccionado debe tener justificante de pago subido.
* Descripción: el usuario administrador podrá validar o no validar el justificante de pago de cualquier usuario registrado de la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de pagos.

2. El usuario administrador modifica el campo de la tabla de la columna “pagado” situado en la fila del usuario elegido.

3. El sistema abre una ventana modal con las acciones posibles.

4. El usuario clica en el botón “Sí”.

5. El sistema actualiza la validación de pago del usuario y notifica al usuario administrador el éxito de la operación.

5. a) Si se valida por primera vez, el sistema asigna número de factura al usuario y habilita los documentos en el perfil del usuario.

5. b) Si se valida y no es la primera vez, el sistema habilita los documentos en el perfil de usuario.

5. c) Si se quita la validación, el sistema deshabilita los documentos en el perfil de usuario, pero mantiene el número de factura para el usuario.

* Excepciones:  
  4. El usuario administrador cierra la ventana modal. El pago no cambia de estado.

4.2.3.8 Descarga de factura de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar la factura de cualquier usuario registrado de la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de pagos.

2. El usuario administrador clica en el icono de la columna “factura” situado en la fila del usuario elegido.

3. El sistema abre una pestaña nueva con el archivo.

4.2.3.9 Visualización tabla de autores

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar la tabla con un resumen de los autores.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de autores.

2. El sistema muestra una tabla resumen con el listado de autores.

4.2.3.10 Visualización tabla de artículos

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar la tabla con un resumen de todos los artículos subidos.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de artículos.

2. El sistema muestra una tabla resumen con el listado de todos los artículos subidos a la aplicación.

4.2.3.11 Visualización artículo de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y al menos un usuario debe haber subido un artículo.
* Descripción: el usuario administrador podrá visualizar el artículo de cualquier usuario registrado de la aplicación que haya subido el archivo.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de artículos.

2. El usuario administrador clica en el icono de la columna “artículo” situado en la fila del artículo elegido.

3. El sistema abre una pestaña nueva en el navegador con el archivo.

4.2.3.12 Modificación estado de artículo de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y al menos un usuario debe haber subido un artículo.
* Descripción: el usuario administrador podrá modificar el estado de cualquier artículo subido a la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de artículos.

2. El usuario administrador modifica el campo de la columna “estado” situado en la fila del artículo elegido.

3. El sistema guarda el cambio de estado del artículo en la base de datos y notifica al usuario administrador el éxito en la operación.

4. El sistema envía un correo electrónico al usuario que subió el artículo y al usuario administrador indicando el cambio de estado del artículo.

4.2.3.13 Modificación artículo de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y al menos un usuario debe haber subido un artículo.
* Descripción: el usuario administrador podrá modificar los datos de cualquier artículo subido a la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de artículos.

2. El usuario administrador clica en el icono izquierdo de la columna “acciones” del artículo a modificar.

3. El sistema abre una ventana modal con los datos del artículo.

4. El usuario administrador modifica los datos del artículo que desee cambiar.

5. El usuario clica en el botón “Guardar cambios”.

6. El sistema actualiza los datos del artículo en la base de datos, cierra la ventana modal y notifica al usuario el éxito de la operación.

* Excepciones:  
  5. El usuario administrador cierra la ventana modal. Los datos modificados no se guardan.

4.2.3.14 Eliminar artículo de usuario

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión y al menos un usuario debe haber subido un artículo.
* Descripción: el usuario administrador podrá eliminar cualquier artículo subido a la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de artículos.

2. El usuario administrador clica en el icono derecho de la columna “acciones” del artículo a modificar.

3. El sistema abre una ventana modal con las acciones posibles.

4. El usuario administrador clica en el botón “Sí”.

5. El sistema elimina el artículo de la base de datos, cierra la ventana modal y notifica al usuario el éxito de la operación.

* Excepciones:  
  5. El usuario administrador cierra la ventana modal. El artículo no se elimina.

4.2.3.15 Visualización tabla de conferencias

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá visualizar la tabla con todas las conferencias guardadas en el sistema.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de conferencias.

2. El sistema muestra una tabla con el listado de todas las conferencias disponibles en la aplicación.

4.2.3.16 Añadir nueva conferencia

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá añadir una nueva conferencia a la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de conferencias.

2. En la sección “Nueva conferencia”, el usuario añade un título para la nueva conferencia.

3. El usuario administrador clica en el botón “Añadir conferencia”.

4. El sistema añade a la base de datos la nueva conferencia, actualiza la tabla en la pantalla conferencias y notifica al usuario administrador el éxito en la operación.

* Excepciones:  
  3. El usuario administrador cambia de pantalla o sale de la aplicación. La conferencia no se añade.

4.2.3.17 Modificación conferencia

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo una conferencia en la aplicación.
* Descripción: el usuario administrador podrá modificar una conferencia ya existente en la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de conferencias.

2. El usuario clica en el icono izquierdo (editar) de la columna “acciones” de la conferencia a la que va a modificar.

3. El sistema abre una ventana modal con el título de la conferencia.

4. El usuario modifica el título.

5. El usuario clica en el botón “Guardar cambios”.

4. El sistema modifica en la base de datos el título de la conferencia, actualiza la tabla en la pantalla conferencias y notifica al usuario administrador el éxito en la operación.

* Excepciones:  
  5. El usuario administrador cierra la ventana modal. La conferencia no se modifica.

4.2.3.18 Eliminar conferencia

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo una conferencia en la aplicación.
* Descripción: el usuario administrador podrá eliminar una conferencia ya existente en la aplicación.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de conferencias.

2. El usuario clica en el icono derecho (borrar) de la columna “acciones” de la conferencia a la que va a eliminar.

3. El sistema abre una ventana modal con las opciones posibles.

4. El usuario clica en el botón “Sí”.

5. El sistema elimina de la base de datos la conferencia, actualiza la tabla en la pantalla conferencias y notifica al usuario administrador el éxito en la operación.

* Excepciones:  
  4. El usuario administrador cierra la ventana modal. La conferencia no se elimina.

4.2.3.19 Descarga de certificados de asistencia

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un usuario registrado en la aplicación.
* Descripción: el usuario administrador podrá descargar en formato PDF todos los certificados de asistencia de los usuarios.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Certificados de asistencia” de la sección “Documentos PDF”.

3. El sistema descarga un archivo PDF con todos los certificados de asistencia.

4.2.3.20 Descarga de certificados de ponencia

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un artículo subido a la aplicación con estado “aceptado”.
* Descripción: el usuario administrador podrá descargar en formato PDF todos los certificados de ponencia de todos los artículos con estado “aceptado”.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Certificados de ponencia” de la sección “Documentos PDF”.

3. El sistema descarga un archivo PDF con todos los certificados de ponencia de todos los artículos en estado “aceptado”.

4.2.3.21 Descarga de facturas

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un usuario registrado.
* Descripción: el usuario administrador podrá descargar todas las facturas de los usuarios registrados en formato PDF.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Facturas” de la sección “Documentos PDF”.

3. El sistema descarga un archivo PDF con todas las facturas de los usuarios registrados.

4.2.3.22 Descarga de lista de usuarios

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un usuario registrado.
* Descripción: el usuario administrador podrá descargar la lista de usuarios en formato Excel.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Usuarios” de la sección “Documentos Excel”.

3. El sistema descarga un archivo Excel con la lista de los usuarios registrados.

4.2.3.23 Descarga de lista de artículos

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un artículo subido.
* Descripción: el usuario administrador podrá descargar la lista de artículos en formato Excel.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Artículos” de la sección “Documentos Excel”.

3. El sistema descarga un archivo Excel con la lista de los artículos subidos.

4.2.3.24 Descarga de lista de autores

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un artículo subido.
* Descripción: el usuario administrador podrá descargar la lista de autores en formato Excel.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario clica en el botón “Autores” de la sección “Documentos Excel”.

3. El sistema descarga un archivo Excel con la lista de los autores de la aplicación.

4.2.3.25 Descarga de etiquetas

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión. Debe haber como mínimo un usuario registrado.
* Descripción: el usuario administrador podrá descargar la lista de etiquetas para todos los usuarios en formato PDF.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de documentos.

2. El usuario rellena los campos de la sección “Etiquetas” a excepción del último campo (no obligatorio).

3. El usuario clica en el botón “Etiquetas”.

3. El sistema descarga un archivo PDF con las etiquetas identificativas de todos los usuarios.

* Flujo alternativo:

2. El usuario rellena todos los campos de la sección “Etiquetas”.

3. El usuario clica en el botón “Etiquetas”.

4. El sistema descarga un archivo PDF con etiquetas que incluyen el texto introducido en el último campo de la sección “Etiquetas”.

4.2.3.26 Modificación de la configuración

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá modificar la configuración de la conferencia.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de configuración.

2. El usuario clica en el botón “Cambiar datos”.

3. El sistema habilita los campos para que puedan ser modificados.

4. El usuario modifica los campos correspondientes.

5. El usuario administrador clica en el botón “Guardar cambios”.

6. El sistema actualiza la base de datos con la configuración modificada, notifica al usuario administrador el éxito de la operación y vuelve a deshabilitar los campos de la pantalla.

* Excepciones:  
  5. El usuario administrador cambia de pantalla o sale de la aplicación. Los datos de la configuración no son modificados.

4.2.3.27 Resetear base de datos

* Actores: usuario administrador.
* Precondiciones: el usuario administrador debe haber iniciado sesión.
* Descripción: el usuario administrador podrá eliminar los datos de la base de datos.
* Flujo normal del caso de uso:

1. El usuario administrador accede a la pantalla de configuración.

2. El usuario clica en el *toggle* “Habilitar botones”.

3. El sistema habilita los botones de reseteo de la base de datos.

4. El usuario clica en los botones “Resetear usuarios” y “Resetear conferencias”.

5. El sistema elimina de la base de datos todos los usuarios, todas las conferencias y todos los artículos. Además, notifica al usuario administrador el éxito de la operación.

* Flujo alternativo:

4. El usuario administrador clica en el botón “Resetear usuarios”.

5. El sistema elimina de la base de datos todos los usuarios y todos los artículos. Además, notifica al usuario administrador el éxito de la operación.

* Flujo alternativo:

4. El usuario administrador clica en el botón “Resetear conferencias”.

5. El sistema elimina de la base de datos todas las conferencias. Además, notifica al usuario administrador el éxito de la operación.

* Excepciones:  
  4. El usuario administrador cambia de pantalla o sale de la aplicación. La base de datos no será reseteada.

4.3 Requisitos no funcionales

Los requisitos no funcionales se centran en aspectos como la eficiencia, seguridad, rendimiento, usabilidad o accesibilidad. La aplicación debe garantizar estos requisitos para obtener la mayor calidad posible en el uso del sistema. A continuación, se detallan estos requisitos:

1. La aplicación podrá ser usada en cualquier navegador web o móvil.
2. La aplicación tendrá que ser responsiva, es decir, deberá adaptarse visualmente a cualquier dispositivo, ya sea web o móvil.
3. El sistema deberá ser internacionalizado. Los usuarios de habla inglesa y castellana podrán disfrutar de una versión en su idioma nativo.
4. La aplicación debe mantener una interfaz similar en todas las pantallas, con los mismos colores, formas y tipos de letras.
5. El sistema tendrá que almacenar las contraseñas en la base de datos con un método de encriptación simétrica.
6. La aplicación tendrá que sacar de sesión a los usuarios a los 30 minutos de inactividad.
7. Los tiempos de espera para la carga y ejecución de la aplicación deben ser inferiores a 10 segundos.
8. La aplicación web contará con un certificado SSL que garantice que es un sitio seguro.
9. La aplicación no podrá tener enlaces rotos. El usuario podrá navegar por la aplicación sin encontrar pantallas que no le permitan seguir navegando.
10. Cualquier persona, sin importar su edad, sexo, condición social o cultura podrá usar la aplicación.
11. La aplicación tendrá que ser amigable con el usuario, fácil de usar y fácil de aprender.
12. La aplicación deberá indicar al usuario cualquier fallo o éxito en las operaciones que realice.
13. El sistema debe ser autónomo, los propios usuarios deben tener el control sobre la aplicación.
14. La aplicación tendrá reversibilidad, es decir, permitirá deshacer cualquier acción realizada por el usuario, ya sea un error o un cambio futuro.

4.4 Modelo de datos

En este apartado se expondrá el modelo de datos de la aplicación. El objetivo del diseño del modelo de datos es representar la información del sistema de forma ordenada y óptima. A continuación, se detallará la descripción de los modelos de la aplicación.

4.4.1 Descripción de los modelos

Para representar correctamente la estructura del modelo de datos de la aplicación, se hará una descomposición de las colecciones de documentos en las que está formada la base de datos no relacional del sistema. Algunas de las colecciones forman parte de otras colecciones. En primer lugar, se detallarán las colecciones padre, que no forman parte de otra colección. Después se mostrarán las colecciones hijas o subcolecciones.

4.4.1.1 Colección *Users*

Se trata de la colección que almacena toda la información relativa a los usuarios registrados en la aplicación. Cada vez que se registra un usuario, se añadirá un elemento a esta colección. La siguiente tabla muestra en detalle la colección de usuarios del sistema:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| address | string | Dirección del usuario |
| attendance\_downloaded | boolean | Estado del certificado de asistencia descargado |
| bill | collection | Información relativa a la factura |
| check\_payment | boolean | Estado del justificante de pago |
| city | string | Ciudad del usuario |
| country | string | País del usuario |
| email | string | Email del usuario |
| first\_name | string | Nombre del usuario |
| id | string | ID del usuario |
| invoice\_downloaded | boolean | Estado de la factura descargada |
| last\_name | string | Apellidos del usuario |
| papers | collection | Colección que recoge los artículos del usuario |
| password | string | Contraseña cifrada del usuario |
| payment\_file | collection | Información relativa al justificante de pago |
| phone | number | Número de teléfono del usuario |
| postal\_code | string | Código postal del usuario |
| presentation\_downloaded | boolean | Estado de certificado de presentación descargado |
| state | string | Estado o comunidad del usuario |
| tax | number | Tarifa a pagar por el usuario |
| title | string | Título del usuario |
| university\_company | string | Universidad, compañía o colegio |

Tabla 4 Colección Users

4.4.1.2 Colección *Config*

Es la colección que almacena toda la información relativa a la configuración de la aplicación. Esta información solo será accesible para el usuario administrador. La siguiente tabla muestra en detalle la colección de configuración del sistema:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| CIF | string | CIF del CMMSE |
| ISBN | string | Número ISBN del CMMSE |
| ISSN | string | Número de ISSN del CMMSE |
| bill\_number | number | Número de factura a asignar |
| certificate\_signature | date | Fecha de firma de certificados |
| cmmse\_opened | boolean | Estado del congreso |
| conference\_end\_day | date | Fecha de fin del congreso |
| conference\_initial\_day | date | Fecha de comienzo del congreso |
| conference\_place | string | Lugar del congreso |
| conference\_url | string | Enlace a la web del congreso |
| conference\_year | number | Año del congreso |
| email\_opened | boolean | Estado de gestión de envío de correos |
| email\_password | string | Contraseña del correo |
| email\_sender | string | Dirección de envío de correo |
| email | array | Conjunto de direcciones en copia de correos |
| fee\_to\_pay | number | Tarifa estándar para facturas |
| id | string | ID de la configuración |
| invoice\_signature | date | Fecha de firma de facturas |
| root\_password | string | Contraseña cifrada del usuario administrador |
| root\_user | string | Nombre de usuario del usuario administrador |
| send\_term\_opened | boolean | Estado de subida de artículos al sistema |

Tabla 5 Colección Config

4.4.1.3 Colección *Conferences*

La colección *Conferences* es la que almacena todas las conferencias del sistema. El usuario administrador podrá añadir una nueva conferencia a la colección para que los usuarios puedan indicar al subir un artículo que ese artículo pertenece a esa conferencia. La siguiente tabla muestra en detalle la colección de conferencias del sistema:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| id | string | ID de la conferencia |
| value | string | Título de la conferencia |

Tabla 6 Colección Conferences

4.4.1.4 Colección *Bill*

La colección *Bill* es una subcolección de *Users*. En ella se almacenan los datos relativos a la factura del usuario. La siguiente tabla muestra en detalle la colección de factura de los usuarios:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| CIF | string | CIF para la factura |
| address | string | Dirección de facturación |
| first\_name | string | Nombre para la factura |
| invoice\_number | number | Número de factura |
| last\_name | string | Apellido para la factura |
| title | string | Título para la factura |
| university\_company | string | Universidad, compañía o colegio para la factura |

Tabla 7 Colección Bill

4.4.1.5 Colección *Papers*

La colección *Papers* es una subcolección de *Users*. En ella se almacenan los datos relativos a los artículos subidos por el usuario. La siguiente tabla muestra en detalle la colección de artículos de los usuarios:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| authors | collection | Información relativa a los autores |
| id\_file | string | ID del archivo del artículo |
| date | date | Fecha de subida del artículo |
| minisymposium | string | Título de la conferencia a la que pertenece |
| name\_file | string | Nombre del archivo del artículo |
| poster | boolean | Valor que indica si es el artículo es un poster |
| state | string | Estado del artículo |
| title | string | Título del artículo |
| url\_file | string | Dirección del archivo en Firebase |

Tabla 8 Colección Papers

4.4.1.6 Colección *Payment\_file*

La colección *Payment\_file* es una subcolección de *Users*. En ella se almacenan los datos relativos al justificante de pago subido por el usuario. La siguiente tabla muestra en detalle esta subcolección:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| id\_file | string | ID del archivo del justificante de pago |
| name\_file | string | Nombre del archivo del justificante de pago |
| url\_file | string | Dirección del archivo en Firebase |

Tabla 9 Colección Payment\_file

4.4.1.7 Colección *Authors*

La colección *Authors* es una subcolección de *Papers*. En ella se almacenan los datos relativos a los autores de cada artículo subido por el usuario. La siguiente tabla muestra en detalle esta subcolección:

|  |  |  |
| --- | --- | --- |
| Atributos de la colección | Tipo | Descripción |
| author | number | Número de autor del artículo |
| email | string | Dirección email del autor |
| first\_name | string | Nombre del autor |
| last\_name | string | Apellidos del autor |

Tabla 10 Colección Authors

Una vez vistas todas las colecciones existentes, en la siguiente figura se detalla un diagrama completo de la base de datos (Cloud Firestore). Dentro de Firebase, se pueden crear proyectos independientes entre sí. El proyecto creado en Firebase para alojar la aplicación se llama CMMSE. Del proyecto, en la base de datos se alojan las colecciones tal como se indica en el siguiente diagrama.

Figura 26 Diagrama estructural de la base de datos

# DOCUMENTO 5: DISEÑO DEL SISTEMA

**D. VIGIL RODRÍGUEZ, Guillermo**

**TUTOR: D. RANILLA PASTOR, José**

**COTUTOR: D. REDONDO LÓPEZ, José Manuel**

**FECHA: Julio 2018**

## 5.1 Introducción

En este apartado nos centraremos en el diseño de la aplicación web, describiendo los diferentes diagramas (de paquetes, clases, de interacción) esenciales, así como el diseño de la base de datos y los diferentes diseños preliminares y definitivos de las interfaces de usuario, lo que ayudará a comprender mejor el proceso de diseño que se ha seguido en el desarrollo de esta aplicación web.

5.2 Diagrama de paquetes

Como ya hemos mencionado y descrito anteriormente, nuestra aplicación web sigue el patrón de diseño MVC (Modelo-Vista-Controlar), por lo que los distintos módulos de la aplicación van a estar claramente diferenciados y separados lógicamente.

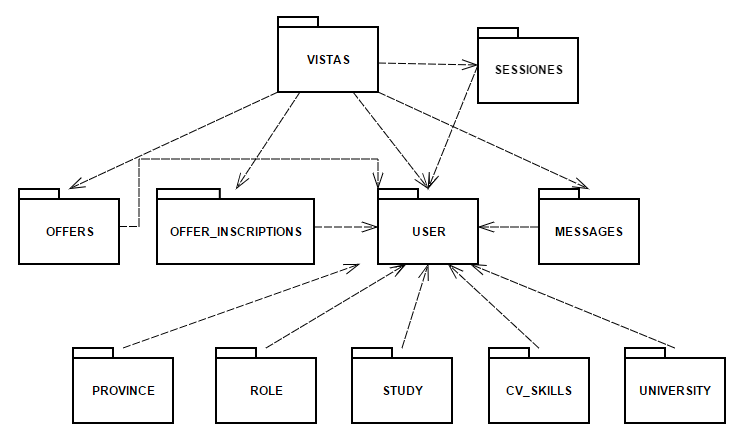


Figura 27 Diagrama de paquetes del sistema

A continuación se hará una breve descripción de los paquetes más importantes de esta aplicación:

5.2.1 User

Este paquete engloba las vistas, modelo y controlador de usuario, los cuales son necesarios para la gestión de estos (registrar usuarios, configuración de cuenta, configuración de perfil profesional, etc). Los usuarios podrán tendrán asociadas (dependiendo del rol que tenga) varias competencias de usuario (cv\_skills), una provincia (province), unos estudios principales (study), un rol (role) y una universidad (university). También puede tener asociadas varias ofertas de prácticas (offers) o participar en varios procesos de selección (offer\_inscriptions). Así mismo los usuarios pueden recibir o enviar varios mensajes (messages) diferentes.

5.2.2 Vistas

Este paquete forma la parte gráfica de la aplicación web, es decir, la capa de presentación de la aplicación web. Las vistas se encargan de recibir los datos de la aplicación enviados desde el controlador para mostrárselos al usuario.

5.2.3 Offers

Este paquete engloba las vistas, modelo y controlador de las ofertas de prácticas, las cuales son necesarios para la creación y gestión de estas. Las ofertas de prácticas están asociadas a un usuario con rol de empresa y tienen asociadas un proceso de selección (offer\_inscription).

5.2.4 Offer\_inscriptions

Este paquete engloba las vistas, modelo y controlador de los procesos de selección pertenecientes a las ofertas de prácticas, las cuales son necesarios para la gestión de estos. Estos procesos de selección están asociados a una oferta de prácticas y a un usuario con rol de empresa. Cada proceso de selección se especificará en diferentes fases mediante intervalos de tiempo fijados por el sistema.

5.2.5 Messages

Este paquete engloba las vistas, modelo y controlador de la gestión de mensajes entre usuarios. Un usuario podrá enviar un mensaje a un usuario así como recibir todos aquellos mensajes que le envíen otros usuarios.

5.2.6 Sessions

Es el paquete encargado de gestionar las sesiones de usuario en la aplicación web. Se encarga de guardar en base de datos las sesiones iniciadas de cada usuario y de validar si la sesión es la correcta mediante el identificador único que posee cada sesión.

5.3 Diagrama de clases

En el apartado “4.4 Modelo de datos” del documento 4 “Análisis de requisitos del sistema” definimos el modelos de datos que debería seguir la aplicación web mediante la descripción individual de cada modelo y posteriormente con un diagrama entidad-relación que relacionase estos. Este apartado nos dio una idea general y bastante cercana del modelo de datos que debe seguir la aplicación y por consiguiente sus correspondientes clases, ya que estas se encargan de representar a las tablas de la base de datos.

A continuación se expondrá un diagrama de clases generado con la herramienta RailRoady, la cual sirve para generar automáticamente y de forma muy sencilla diagramas de clases de cualquier aplicación web que se desarrolle en Ruby on Rails. Esta herramienta proporciona un archivo “.svg” que puede ser leído por cualquier navegador o programa de tratamiento de imágenes. A continuación se adjunta el diagrama generado de esta aplicación web:



Figura 28 Diagrama de clases de la aplicación.

5.4 Diagrama de interacción

En este apartado se presentará el diagrama de interacción del sistema el cual define como se va a llevar a cabo la interacción del usuario con el sistema de forma general. Este diagrama se verá fuertemente influenciado por el patrón MVC, ya que cada vez que el usuario interactúe con el sistema éste seguirá el diseño del patrón MVC en el tratamiento de la información requerida por el usuario.

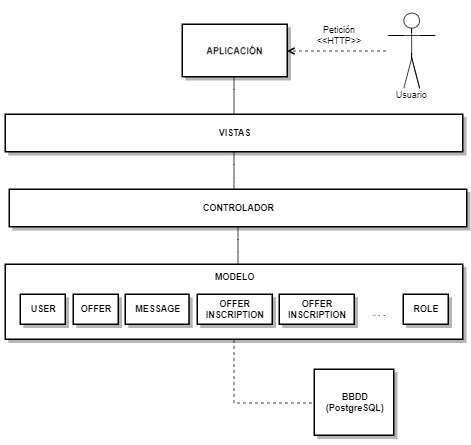


Figura 29 Diagrama de interacción sistema-usuario

Cuando el usuario realice algún tipo de petición HTTP a la aplicación web a través de cualquiera de sus vistas el controlador será quién reciba la notificación del evento hecho por el usuario, encargándose posteriormente de gestionarlo.

El controlador seguidamente accederá al modelo, ya sea para crear o modificar en él algún dato pedido por el usuario (crear oferta de prácticas, modificar competencia profesional, etc) o bien para solicitar información de algún modelo (ver perfil de usuario estudiante, ver información de una oferta de prácticas, etc.). El modelo es quién se encarga de gestionar los datos de la aplicación, sin embargo este no tiene conocimiento directo sobre las vistas.

Por último y una vez obtenidos los datos necesarios del modelo, el controlar delega toda la información obtenida a las correspondientes vistas que deberán generar la interfaz apropiada para el usuario.

5.5 Diseño de la bases de datos

A continuación se hará una descripción del sistema de gestión usado para la base de datos de la aplicación así como la integración de esta en nuestro sistema.

5.5.1 Sistema Gestor de la Base de Datos

Como ya hemos mencionado anteriormente el sistema gestor escogido que administrará la base de datos PostgreSQL de nuestra aplicación es el pgAdmin III, el cual proporciona una librería gráfica multiplataforma lo que permite usarse en Linux, Windows, Mac OS, Solaris y muchos más sistemas operativos.

Este gestor tiene multitud de características, desde permitir al usuario escribir consultas SQL simples hasta desarrollar bases de datos complejas. También incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados y muchas más características.

Sin duda con este gestor y su multitud de características garantizaremos una buena gestión de nuestra base de datos.

5.5.2 Integración del SGBD en nuestro sistema

A continuación se mostrará la base de datos de nuestra aplicación web, llamada tfg\_v1\_development y la cual estará compuesta por las tablas correspondientes a los modelos que hemos visto en los apartados anteriores. Uno de los archivos principales de todo proyecto en Ruby on Rails es el “schema.rb” archivo el cual representa en código fuente la estructura de la base de datos de nuestro proyecto. Gracias a este archivo tendremos una visión actualizada de nuestra base de datos en todo momento, hagamos cambios en ella o no.

ActiveRecord**::**Schema**.**define**(**version: 20170620162038**)** **do**

# These are extensions that must be enabled in order to support this database

enable\_extension "plpgsql"

create\_table "cv\_skills"**,** force: :cascade **do** **|**t**|**

t**.**integer "user\_id"

t**.**string "nombre"

t**.**string "description"

t**.**datetime "date"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

t**.**index **[**"user\_id"**],** name: "index\_cv\_skills\_on\_user\_id"**,** using: :btree

**end**

create\_table "studies"**,** force: :cascade **do** **|**t**|**

t**.**string "name"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

**end**

create\_table "messages"**,** force: :cascade **do** **|**t**|**

t**.**integer "sender\_id"

t**.**integer "recipient\_id"

t**.**datetime "time\_send"

t**.**string "subject"

t**.**string "body"

t**.**boolean "readed"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

**end**

create\_table "offer\_inscriptions"**,** force: :cascade **do** **|**t**|**

t**.**integer "offer\_id"

t**.**integer "user\_id"

t**.**integer "users\_id\_phase1"**,** default: **[],** array: **true**

t**.**boolean "phase1"

t**.**datetime "phase1\_date"

t**.**integer "users\_ids\_phase2"**,** default: **[],** array: **true**

t**.**boolean "phase2"

t**.**datetime "phase2\_date"

t**.**datetime "end\_date"

t**.**integer "winner\_id"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

t**.**boolean "ended"

t**.**index **[**"offer\_id"**],** name: "index\_offer\_inscriptions\_on\_offer\_id"**,** using: :btree

t**.**index **[**"user\_id"**],** name: "index\_offer\_inscriptions\_on\_user\_id"**,** using: :btree

**end**

create\_table "offers"**,** force: :cascade **do** **|**t**|**

t**.**integer "user\_id"

t**.**string "name"

t**.**integer "perfil\_id"

t**.**string "description"

t**.**string "duration"

t**.**string "start\_date"

t**.**string "finish\_date"

t**.**integer "hours"

t**.**string "location"

t**.**integer "salary"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

t**.**string "requirements"

t**.**string "finish\_inscription\_date"

t**.**index **[**"user\_id"**],** name: "index\_offers\_on\_user\_id"**,** using: :btree

**end**

create\_table "provinces"**,** force: :cascade **do** **|**t**|**

t**.**string "name"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

**end**

create\_table "roles"**,** force: :cascade **do** **|**t**|**

t**.**string "name"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

**end**

create\_table "universidads"**,** force: :cascade **do** **|**t**|**

t**.**string "name"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

**end**

create\_table "users"**,** force: :cascade **do** **|**t**|**

t**.**string "name"

t**.**string "email"

t**.**string "password\_digest"

t**.**datetime "created\_at"**,** null: **false**

t**.**datetime "updated\_at"**,** null: **false**

t**.**integer "role\_id"

t**.**string "last\_name"

t**.**string "birth\_date"

t**.**integer "province"

t**.**string "city"

t**.**datetime "deleted\_at"

t**.**integer "universidad"

t**.**integer "grado"

t**.**string "description"

t**.**index **[**"role\_id"**],** name: "index\_users\_on\_role\_id"**,** using: :btree

**end**

add\_foreign\_key "cv\_skills"**,** "users"

add\_foreign\_key "offer\_inscriptions"**,** "offers"

add\_foreign\_key "offer\_inscriptions"**,** "users"

add\_foreign\_key "offers"**,** "users"

**end**

5.6 Diseño de la interfaz

A continuación se mostrará detalladamente el diseño de las diferentes interfaces de esta aplicación web, desde los interfaces genéricos hasta los diferentes interfaces diseñados para cada rol de usuario.

Todos las interfaces han sido diseñadas utilizando el lenguaje CSS y aprovechando los diseños que proporciona el framework de diseño Bootstrap.

5.6.1 Interfaces públicos

En primer lugar detallaremos aquellos interfaces que son públicos, es decir, están visibles a cualquier usuario que interactúe con la aplicación sin tener que estar registrado o haber iniciado sesión.

5.6.1.1 Pantalla de bienvenida o home

La pantalla de bienvenida o home ha sido diseñada con el objetivo de dar una imagen sencilla a la vez que moderna al usuario que acceda por primera vez a la aplicación, así como también introducir el objetivo principal de esta y hacia quién va destinada, animando al usuario a registrarse en ella.



Figura 30 Captura de pantalla de la pantalla principal de la aplicación

5.6.1.2 Pantalla de login

La pantalla de login sigue un diseño simple a la vez que limpio, donde el usuario deberá introducir sus credenciales de acceso (email y login) para acceder a la parte privada de la aplicación en el caso de estar registrado.

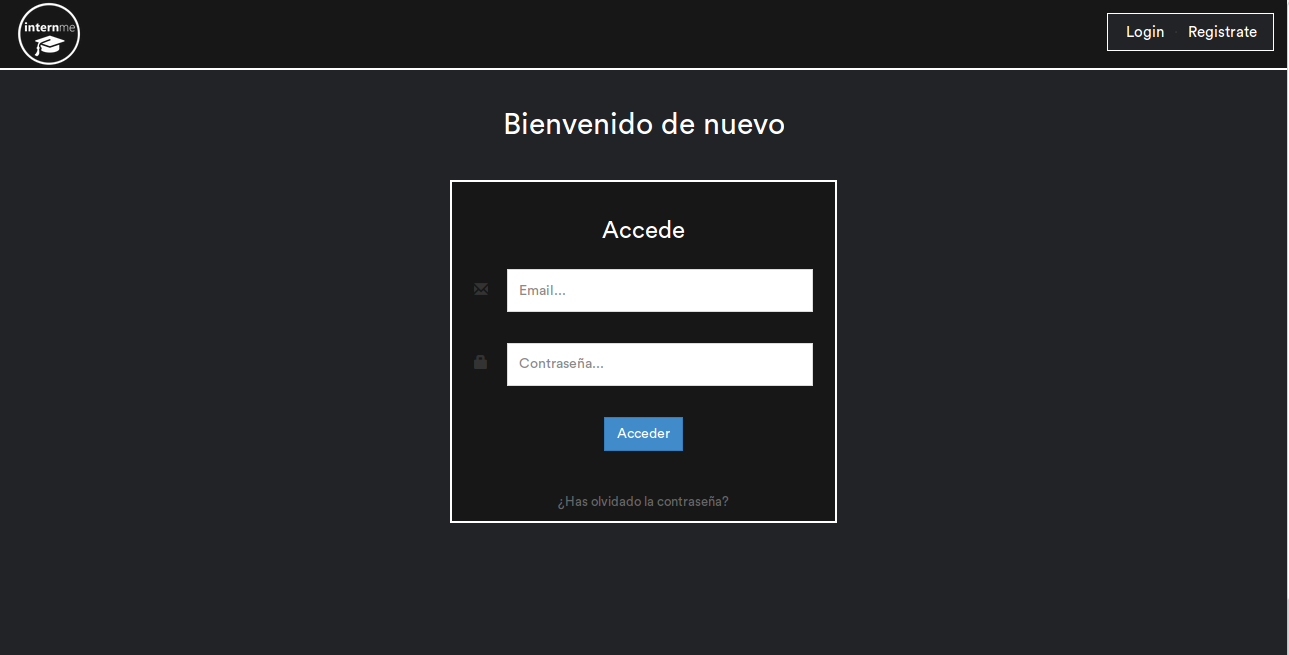


Figura 31 Captura de pantalla de la pantalla de login de la aplicación.

5.6.1.3 Pantalla de recuperar contraseña

A continuación se detallarán las pantallas relativas a la gestión de recuperación de contraseña.



Figura 32 Pantalla de recuperar contraseña 1/2.

Una vez recibido el email con la URL, la vista que se genera para cambiar la contraseña es la siguiente:

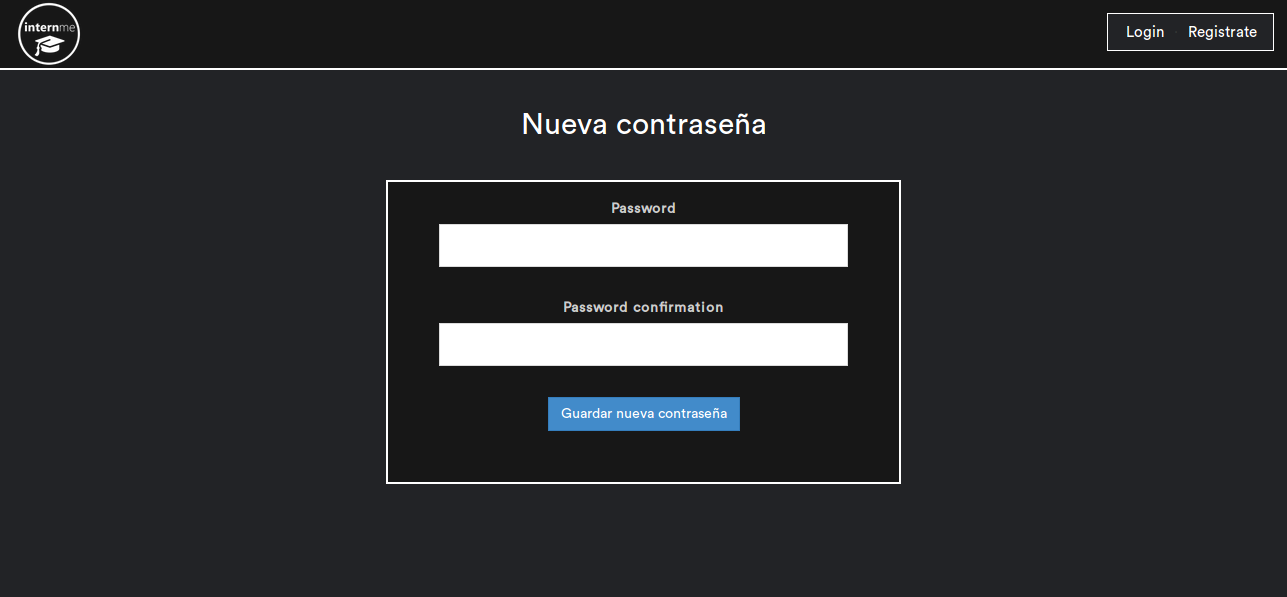


Figura 33Pantalla de recuperar contraseña 2/2.

5.6.1.4 Pantalla de registro

La pantalla principal de registro se ha dividido en dos contenedores con el objetivo de diferenciar el tipo de usuario que se puede registrar en la aplicación web.

****

Figura 34 Captura de pantalla de la pantalla de registro principal de la aplicación

Por un lado tenemos el contenedor izquierdo que hará referencia al formulario de registro de estudiantes y por otro lado tenemos el contenedor derecho que hará referencia al formulario de registro de empresas.

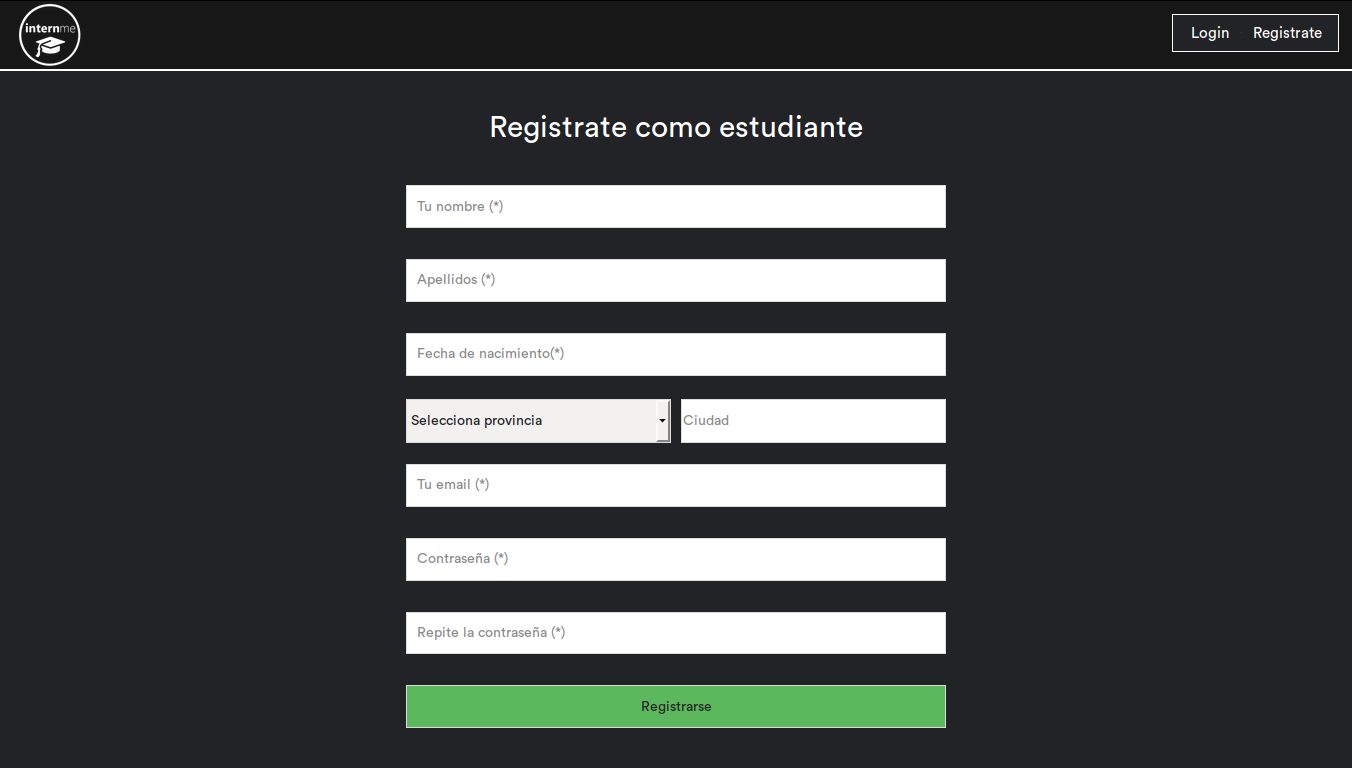


Figura 35 Captura de pantalla de la pantalla de registro de estudiante.

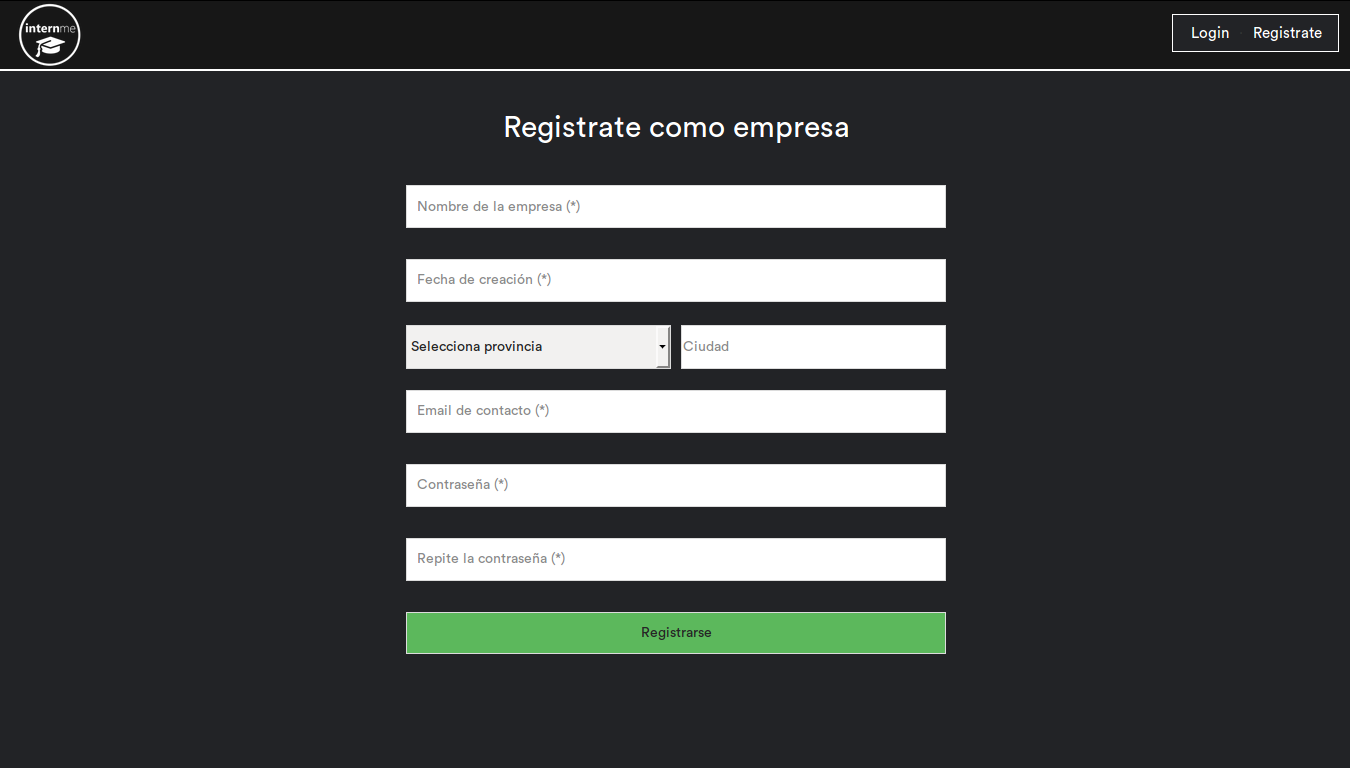


Figura 36 Captura de pantalla de la pantalla de registro de empresa.

5.6.2 Interfaces privados

Una vez vistos los interfaces públicos de la aplicación describiremos aquellos interfaces privados, los cuales solo son visibles por aquellos usuarios que estén registrados en la aplicación y que hayan iniciado sesión.

En este apartado debemos diferenciar aquellos interfaces diseñados para los usuarios con el rol de estudiante y aquellos interfaces diseñados con el rol de empresa, ya que la mayoría de ellos no serán iguales.

5.6.2.1 Interfaces del rol de estudiante

Una de las principales diferencias que encontramos entre los interfaces de estos dos roles es el menú principal de la aplicación, ya que para el rol de estudiante tendrá la opción de acceder a la página principal de empresas y ofertas mientras que para el rol de empresa tendrá la opción de acceder a la página principal de estudiantes y ofertas.

5.6.2.1.1 Pantalla de inicio

En la siguiente imagen podremos ver lo anteriormente expuesto para el rol de usuario, así como el mensaje de confirmación de sesión iniciada que el sistema proporciona cuando el usuario hace login correctamente.

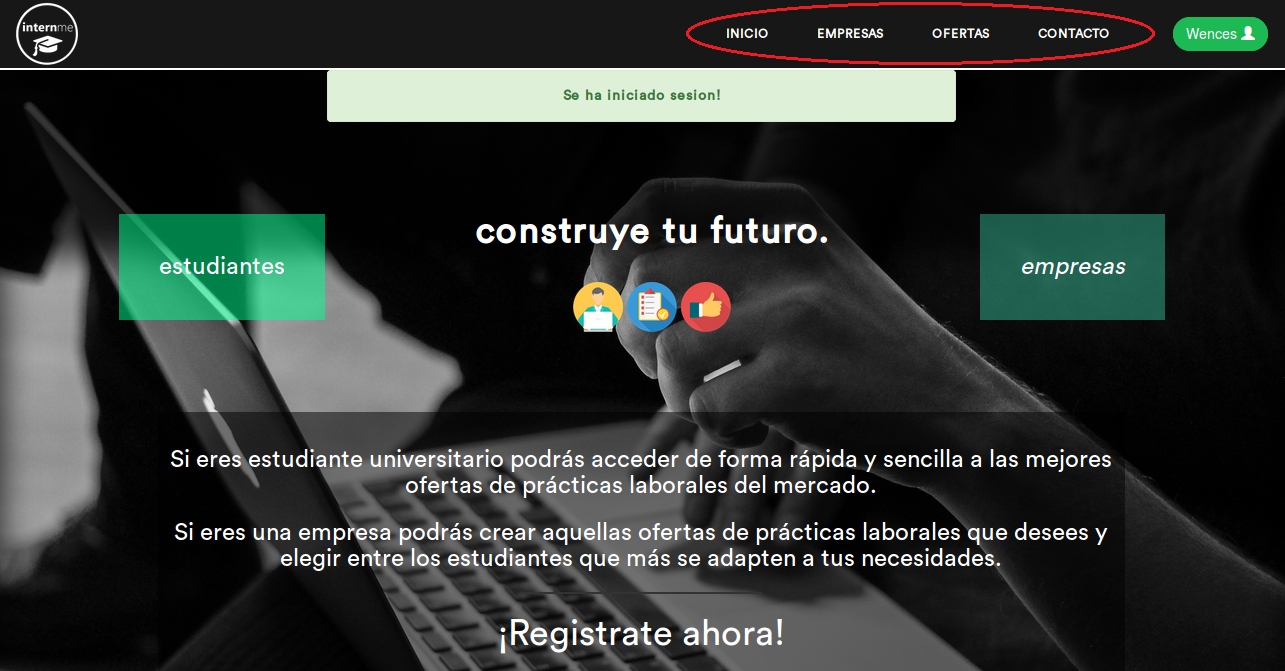


Figura 37 Captura de pantalla de la pantalla de inicio con el rol de estudiante.

5.6.2.1.2 Pantalla de ofertas

El estudiante podrá acceder a la página de ofertas haciendo click en el enlace “Ofertas” ubicado en el menú superior de la página. En ella accederá a todas aquellas ofertas de prácticas de trabajo creadas por las empresas, teniendo la posibilidad de filtrar aquellas ofertas en el buscador por diferentes campos (palabra clave, estudios, provincia o ciudad).



Figura 38 Captura de pantalla de la pantalla de oferta de empresa.

El estudiante podrá ver detalladamente la información completa de cualquier oferta haciendo click en “Ver más” dentro del cuadro de la oferta:



Figura 39 Captura de pantalla de la pantalla de detalle de la oferta seleccionada.

Una vez dentro de la oferta seleccionada, el estudiante puede hacer click en el botón “Inscribirse en la oferta” para inscribirse en esta y participar así en el proceso de selección de esta. A continuación podemos ver en la siguiente imagen que el sistema notifica al usuario por medio de un mensaje que la inscripción se ha realizado correctamente.



Figura 40 Captura de pantalla de la pantalla de detalle de la oferta seleccionada una vez hecha la inscripción.

5.6.2.1.3 Pantalla de empresas y envío de mensajes

Al igual que para las ofertas, el estudiante podrá ver el perfil de todas aquellas empresas registradas en la plataforma haciendo click en el enlace “Empresas” ubicado en el menú superior de la página, así como hacer una búsqueda filtrada (palabra clave, perfil del estudiante buscado, provincia o ciudad) de aquellas empresas que sean de su interés. Por último, el estudiante tendrá la posibilidad de enviar un mensaje a aquellas empresas que desee.

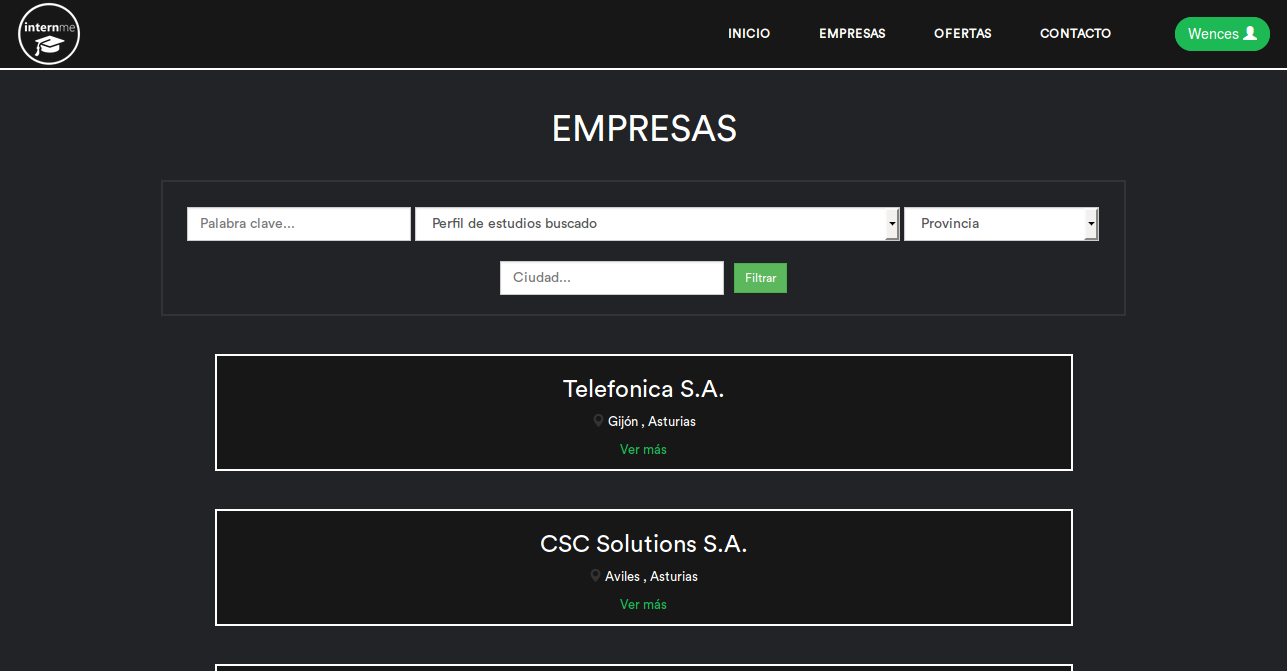


Figura 41 Captura de pantalla de la pantalla principal de empresas con el rol de estudiante.



Figura 42 Captura de pantalla de la pantalla de detalle de la empresa seleccionada.

El diseño que se ha optado para el envío de mensajes ha sido por medio de ventanas modales con peticiones Ajax. Esto se ha hecho con el objetivo de mejorar la experiencia de usuario y hacer la iteración con la página más fluida, ya que con este tipo de peticiones la página no requiere cargarse de nuevo. Cabe destacar que este diseño de interfaces será el mismo tanto para estudiantes como para empresas.

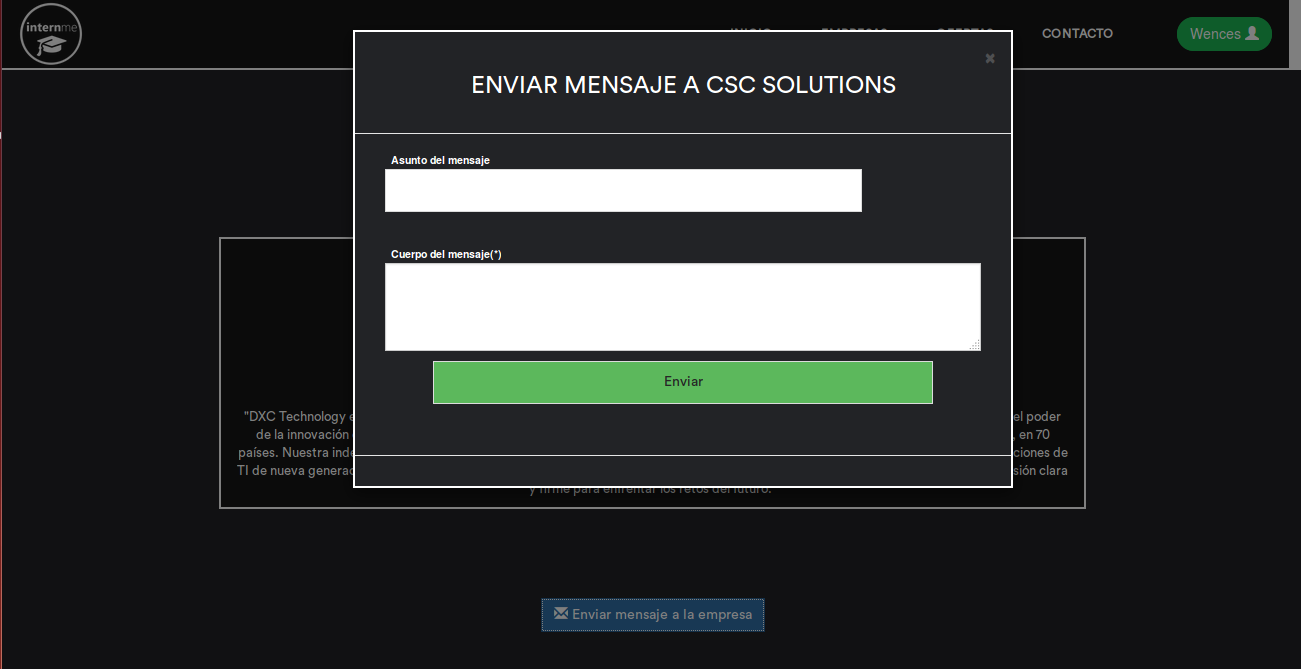


Figura 43 Captura de pantalla de la ventana modal de envío de mensaje.

Una vez enviado el mensaje, el sistema notificará al estudiante del correcto envío del mensaje.

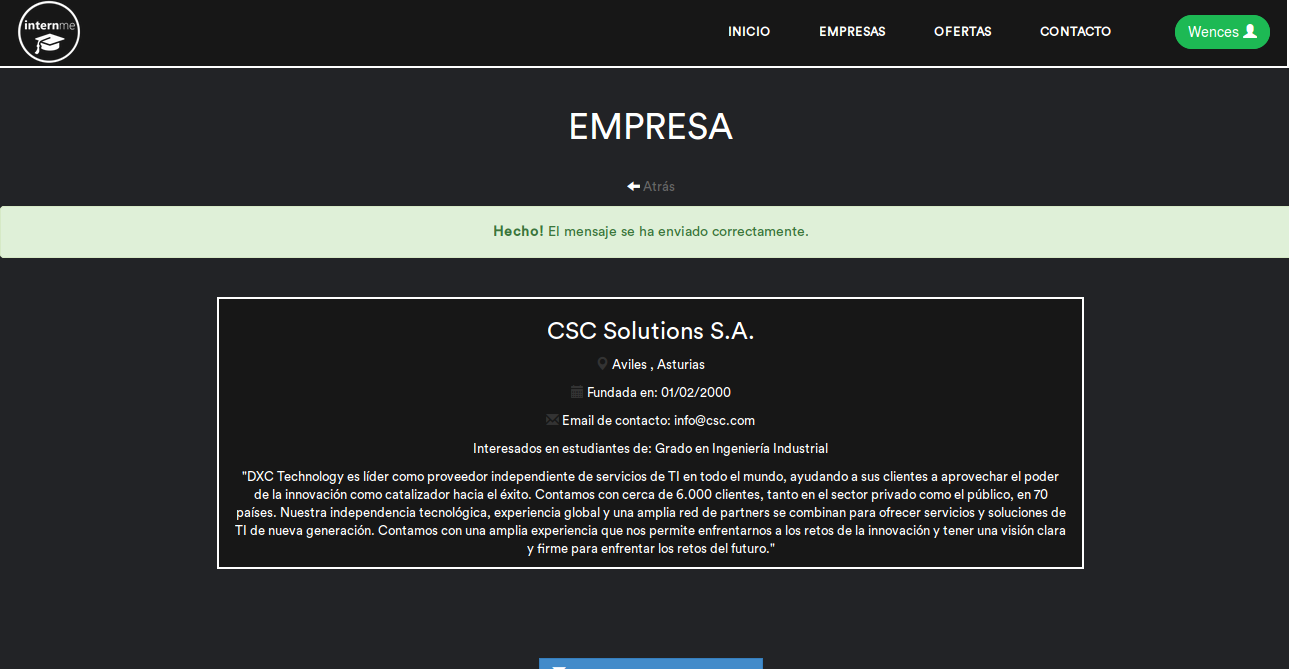


Figura 44 Captura de pantalla de la pantalla de la empresa seleccionada una ve enviado el mensaje.

5.6.2.1.4 Pantalla de perfil

Una de las partes más importantes de la aplicación web es la gestión del perfil del usuario, tanto para estudiantes como para empresas. Se ha decidido hacer un diseño en forma de paneles con sus correspondientes pestañas.

Estos paneles cambiarán a medida que se seleccione una pestaña u otra, permitiendo al usuario navegar entre ellos de forma rápida y sencilla, ya que una vez más la información llevada a los diferentes paneles se ha hecho por medio de peticiones Ajax al servidor, por lo que la página no necesitará recargarse de nuevo cuando cambiemos de pestaña y panel.

5.6.2.1.5 Configuración de cuenta

En este primer panel llamado “Mi perfil” el usuario podrá modificar aquellos datos principales de su cuenta, como su nombre, email, ciudad de origen, edad, contraseña, etc. Se ha utilizado un diseño simple de formulario como los que se han utilizado en el resto de pantallas de la plataforma.



Figura 45 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis Datos”.

5.6.2.1.6 Perfil académico

Este es uno de los paneles más importantes para el estudiante, ya que “Mi CV” es el panel donde el usuario puede definirse profesionalmente e incluir todas aquellas competencias profesionales con las que cuente. Se podría decir que es su curriculum en la aplicación web.

En primer lugar el estudiante podrá seleccionar aquellos estudios universitarios que esté cursando y escribir una breve descripción de sí mismo. A continuación el estudiante podrá poner tantas competencias profesionales como quiera, ya sean otros estudios universitarios realizados con anterioridad, títulos de idiomas, experiencias laborales previas, etc.



Figura 46 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mi CV”.

5.6.2.1.7 Ofertas de usuario

En este tercer panel “Mis ofertas” el estudiante podrá ver un resumen de aquellas ofertas a las que se ha inscrito con anterioridad, así como el estado en el que está cada oferta. Se ha optado por una tabla para representar estos datos.



Figura 47 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis Ofertas”.

5.6.2.1.8 Mensajes

Este último panel “Mis Mensajes” almacenará los mensajes recibidos y enviados por el usuario en la aplicación. El diseño se ha hecho mediante dos tablas, una para los mensajes recibidos, con la posibilidad de leerlos y responderlos así como de ver si ya han sido leídos, y otra tabla para los mensajes que el estudiante ha enviado con anterioridad.



Figura 48 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis Mensajes”.

5.6.2.2 Interfaces del rol de empresa

5.6.2.2.1 Pantalla de inicio

Como mencionamos anteriormente, el rol de empresa tendrá la opción de acceder a la página principal de estudiantes y ofertas en el menú principal.



Figura 49 Captura de pantalla de la pantalla de inicio con el rol de empresa.

5.6.2.2.2 Pantalla de estudiantes y envío de mensajes

La empresa ver el perfil de todos aquellos estudiantes registrados en la plataforma haciendo click en el enlace “Estudiantes” ubicado en el menú superior de la página, así como hacer una búsqueda filtrada (palabra clave, universidad, estudios, provincia o ciudad) de aquellos estudiantes que sean de su interés.

Por último, la empresa (al igual que el estudiante como ya hemos visto) tendrá la posibilidad de enviar un mensaje a aquellos estudiantes que desee.

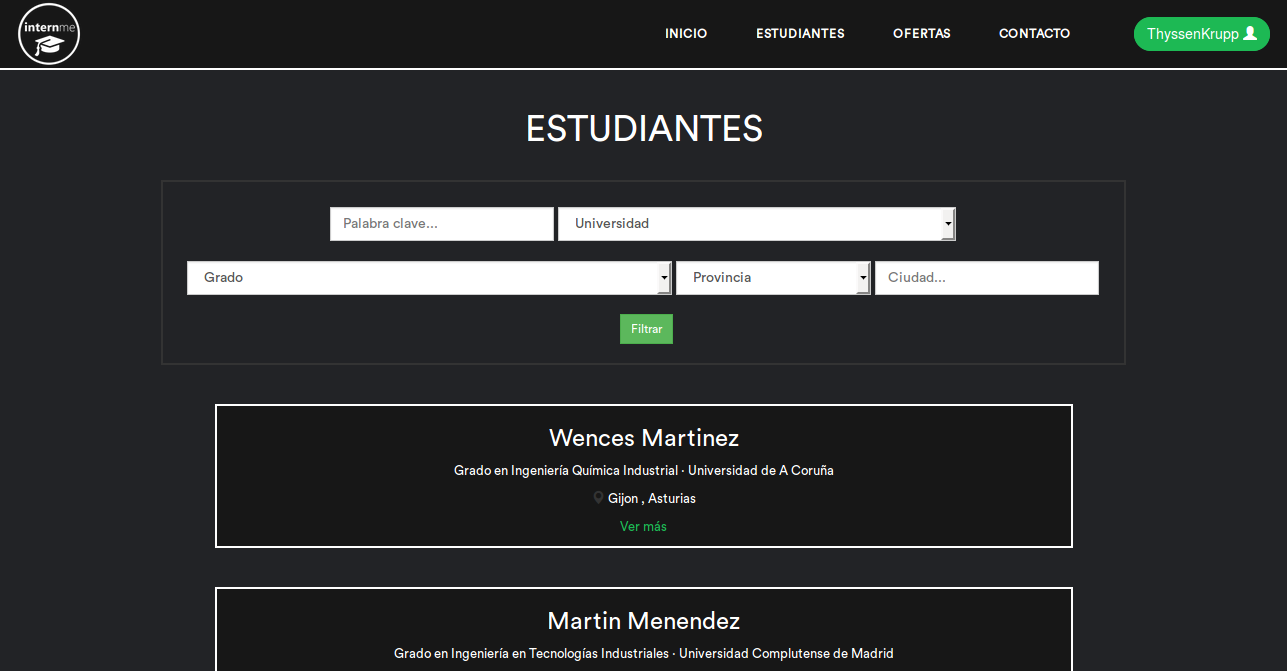


Figura 50 Captura de pantalla de la pantalla principal de estudiantes con el rol de empresa.

La empresa podrá ver detalladamente la información completa de cualquier estudiante haciendo click en “Ver más” dentro del cuadro de este, así como también enviarle un mensaje haciendo click en “Enviar mensaje al estudiante”:



Figura 51 Captura de pantalla de la pantalla de detalle del estudiante seleccionado.

5.6.2.2.3 Pantalla de ofertas

El diseño y funcionamiento de la pantalla de ofertas es exactamente igual tanto para empresas como para estudiantes, por tanto no se pondrán las respectivas interfaces dado que son las mismas que las del apartado XXX del rol de estudiante.

5.6.2.2.4 Pantalla de perfil

Al igual que para el estudiante el apartado “Mi perfil” es uno de los apartados más importantes dentro de la aplicación web para la empresa, ya que desde él podrá gestionar todos los procesos de selección resultantes de las ofertas de prácticas de trabajo que haya creado así como gestionar el envío de mensajes y detallar que perfil de estudiantes busca, así como establecer una breve descripción de la empresa.

5.6.2.2.5 Pantalla de configuración de cuenta

En este primer panel llamado “Mis Datos” el usuario podrá modificar aquellos datos principales de su cuenta, como su nombre, email, ciudad de origen, edad, contraseña, etc. En este caso, la empresa también podrá establecer aquel perfil de estudiante en el que esté interesado y también establecer una breve descripción de la empresa. Se ha utilizado un diseño simple de formulario como los que se han utilizado en el resto de pantallas de la plataforma.



Figura 52 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis datos”.

5.6.2.2.6 Pantalla de ofertas

Al igual que para el estudiante, la podrá ver un resumen de aquellas ofertas que haya creado, pudiendo ver el estado en el que está cada oferta.

Por otra parte y en este mismo apartado, la empresa crear ofertas de prácticas de trabajo haciendo click en el botón “Crear nueva oferta”, lo que cargará un formulario para que la empresa cree una nueva oferta que se publicará en la aplicación una vez rellene los datos requeridos y envíe el formulario.

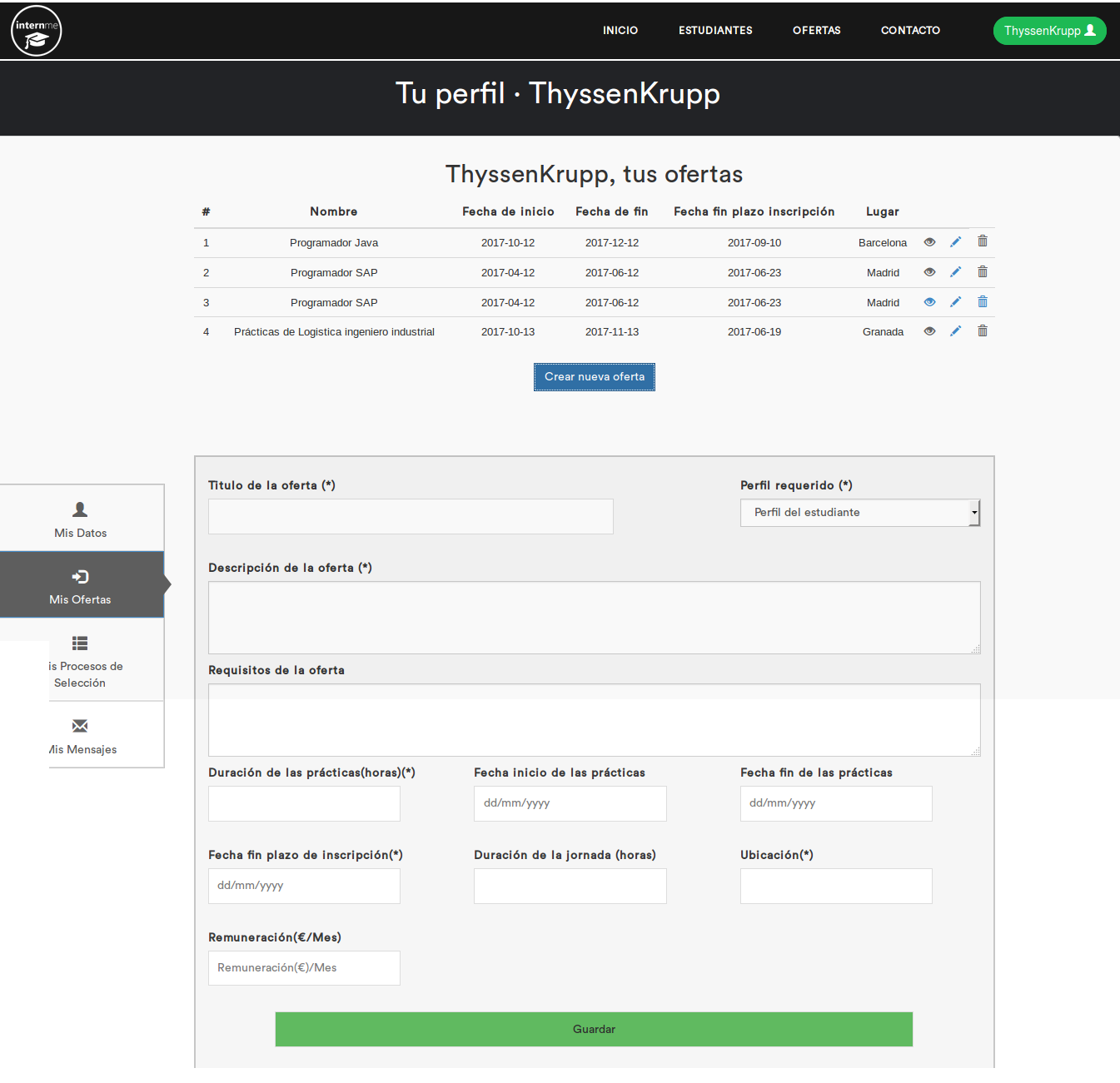


Figura 53 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis ofertas” con el formulario de crear nueva oferta abierto.

5.6.2.2.7 Pantalla de procesos de selección

Este apartado llamado “Procesos de selección” englobará todos aquellos procesos de selección derivados de las ofertas de prácticas que la empresa ha creado y debe gestionar. Como diseño del panel se ha optado por incluir una tabla con todos aquellos procesos de selección y sus diferentes estados, incluyendo la posibilidad de gestionar las respectivas fases de cada proceso de selección.

Cada vez que se haga click en “Gestionar” en alguno de los procesos de selección se nos cargará más abajo un panel que contendrá aquellos estudiantes que estén en la fase actual del proceso de selección, con el objetivo de que la empresa pueda gestionar esa fase del proceso decidiendo entre ellos y pudiendo ver el perfil de cada estudiante en todo momento a través de una ventana modal.

A continuación se muestra un ejemplo de gestión de un proceso de selección con un estudiante inscrito en la oferta de prácticas.

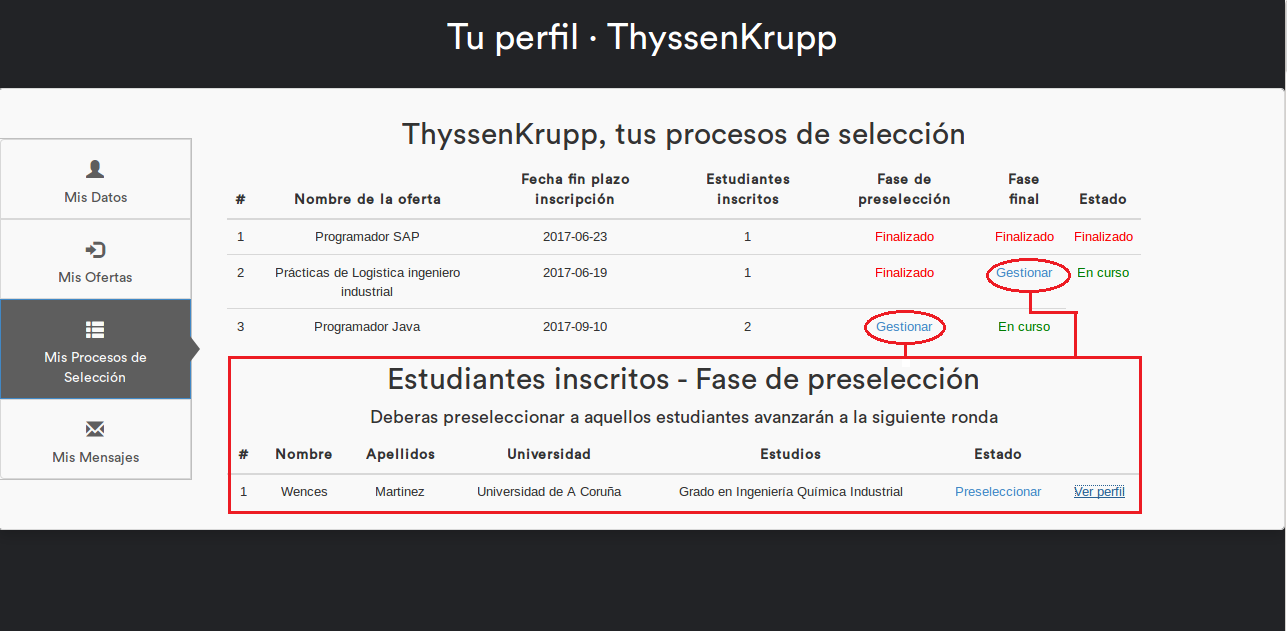


Figura 54 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis procesos de selección” con el contenedor de la fase de preselección de una oferta abierto.

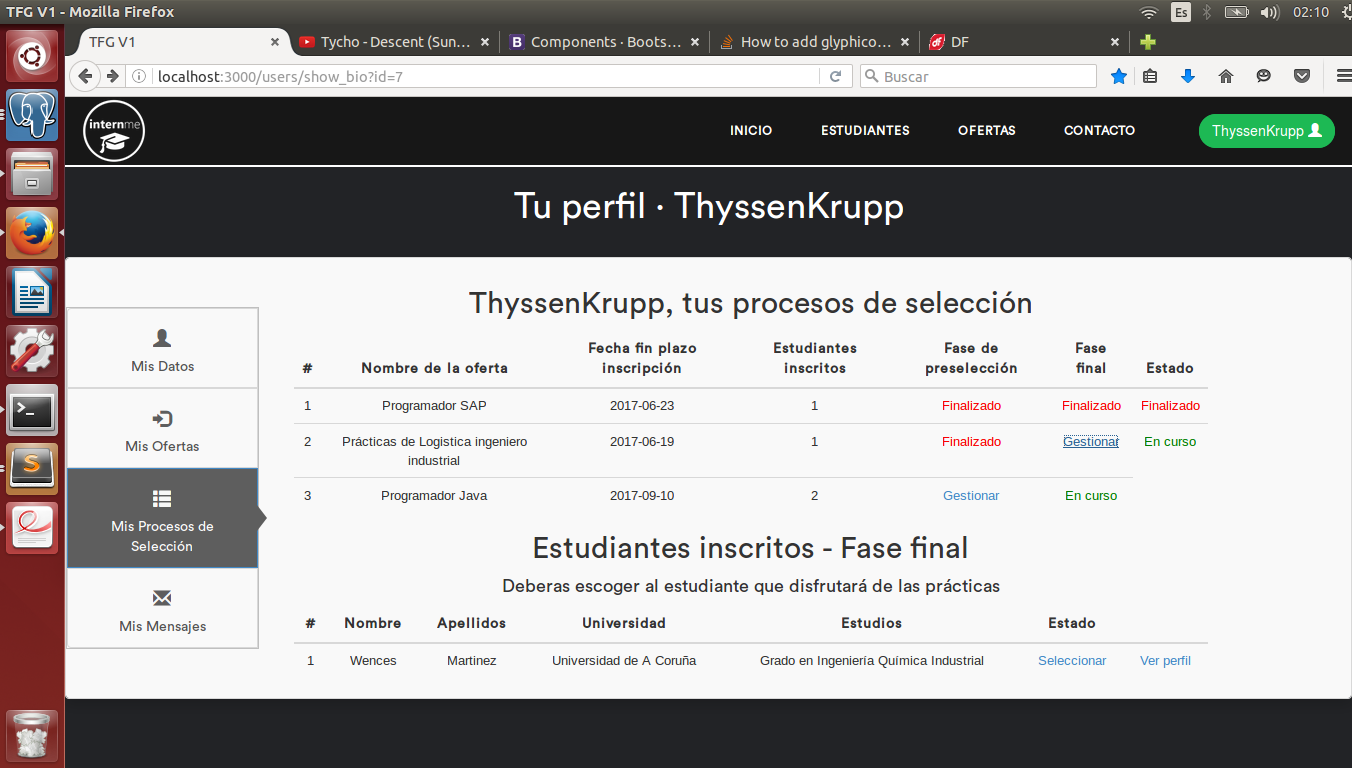


Figura 55 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis procesos de selección” con el contenedor de la fase final de selección de una oferta abierto.

Como podemos ver en la siguiente imagen, la empresa también podrá ver el perfil de aquellos usuarios pertenecientes a las fases del proceso de selección. El diseño de estas interfaces ha seguido una línea prácticamente idéntica a la del envío de mensajes, ya que se ha hecho mediante ventanas modales, lo que hará más fácil y rápida la visualización de los perfiles de los estudiantes, ya que no se cambiará de página en ningún momento.



Figura 56 Captura de pantalla de la pantalla principal de mi perfil en la pestaña “Mis procesos de selección” con la ventana modal de perfil del estudiante seleccionado abierta.

5.6.2.2.8 Pantalla de mensajes

Al igual que para los estudiantes, la pestaña de “Mis Mensajes” está diseñada exactamente igual que como hemos visto anteriormente en el apartado XX para los usuarios con rol de estudiante.

# DOCUMENTO 6: PRUEBAS

**D. MARTÍNEZ SUÁREZ, Wenceslao**

**TUTOR: Dña. SUAREZ CABAL, María José**

**FECHA: Julio 2017**

Índice de contenidos

[**DOCUMENTO 6: PRUEBAS** 120](#_Toc486815244)

[6.1 INTRODUCCIÓN 122](#_Toc486815245)

[6.2 PRUEBAS UNITARIAS 122](#_Toc486815246)

[6.3 PRUEBAS DE INTEGRACIÓN Y DE SISTEMA 124](#_Toc486815247)

[6.4 PRUEBAS DE USABILIDAD Y ACCESIBILIDAD 130](#_Toc486815248)

[6.4.1 CRITERIOS GENERALES 130](#_Toc486815249)

[6.4.2 CRITERIOS DE IDENTIDAD E INFORMACIÓN 131](#_Toc486815250)

[6.4.3 CRITERIOS DE LENGUAJE Y REDACCIÓN 132](#_Toc486815251)

[6.4.4 CRITERIOS DE ROTULADO 133](#_Toc486815252)

[6.4.5 CRITERIOS DE NAVEGACIÓN 133](#_Toc486815253)

[6.4.6 LAY-OUT DE LA PÁGINA 134](#_Toc486815254)

[6.4.7 CRITERIOS DE BÚSQUEDA 135](#_Toc486815255)

[6.4.8 CRITERIOS DE AYUDA 135](#_Toc486815256)

[6.4.9 ACCESIBILIDAD 136](#_Toc486815257)

[6.4.10 CRITERIOS DE CONTROL Y RETROALIMENTACIÓN 137](#_Toc486815258)

[6.5 PRUEBAS DE RENDIMIENTO 137](#_Toc486815259)

## 6.1 Introducción

En este apartado abordaremos todas aquellas pruebas realizadas durante el desarrollo de este proyecto, las cuales han garantizado que el sistema funcione de la mejor manera posible ante las diversas situaciones que se pueden dar, respondiendo tal y como se espera.

El desarrollo de las pruebas se ha dividido en cuatro apartados principales: pruebas unitarias, pruebas de integración y sistema, pruebas de rendimiento y por último pruebas de usabilidad y accesibilidad.

6.2 Pruebas unitarias

Dentro de un sistema con un diseño estructurado, las pruebas unitarias son aquellas pruebas realizadas para las diferentes funcionalidades de cada unidad del sistema, garantizando que estas funcionen correcta y eficientemente por separado.

Para abordar el desarrollo de este tipo de pruebas utilizaremos el TDD o Test Driven Development, disciplina de desarrollo de software en la cual los desarrolladores escriben multitud de casos de prueba para los requerimientos funcionales de una aplicación. El TDD tiene como objetivo garantizar un código limpio, simple y testeable, asegurando la integridad de las diferentes funcionalidades de la aplicación. Una de las mejoras formas de aplicar el TDD a nuestra aplicación en Ruby on Rails es haciendo uso de la gema Rspec, librería creada para realizar tests unitarios para los modelos y test funcionales para los controladores.

Una vez instalada la librería en el proyecto, se debe hacer una configuración inicial de esta para llevar a cabo las pruebas. Se creará una base de datos de pruebas que seguirá la misma estructura que tenemos en la base de datos principal de la aplicación, con las mismas tablas. El hecho de que la base de datos de desarrollo y de pruebas sean iguales nos garantizará que las pruebas llevadas a cabo se asemejarán a la configuración real que tenemos en nuestro proyecto.

Una de las ventajas que tiene Ruby on Rails relacionadas con las pruebas es que esta funcionalidad ya la tiene integrada en el sistema, ya que siempre que se crea un proyecto nuevo el framework incluye una carpeta llamada “test” la cual almacenará todas aquellas pruebas que el desarrollar realice. En la carpeta “test” habrá otra carpeta llamada “models” destinada a albergar las pruebas para los modelos de la aplicación y otra carpeta llamada “controllers” donde se establecerán aquellas pruebas para los controladores.

En el caso de nuestro proyecto nos interesa hacer pruebas de modelos las cuales nos servirán para corregir todos aquellos fallos relacionados con la creación de estos y la modificación de su información. A continuación se detallará un ejemplo de prueba para el modelo “offer”.

El código que se puede ver a continuación se corresponde con un extracto de nuestro modelo “offer”, donde podremos ver que contiene los campos *name, description, perfil\_id, duration, location y finish\_inscription\_date* con validación de presencia activada, lo que quiere decir que nunca se creará un objeto Offer si alguno de estos campos llega vacío.

**class** **Offer** **<** ApplicationRecord

#Libreria de búsqueda PgSearch

include PgSearch

pg\_search\_scope :search\_by\_description**,** :against **=>** **[**:name**,** :description**,** :location**],** using: **{** tsearch: **{** any\_word: **true** **}** **}**

#Relaciones

belongs\_to :user

has\_many :offer\_inscription**,** dependent: :destroy

# Restricciones de validación

validates :name**,** :description**,** :perfil\_id**,** :duration**,** :location**,** :finish\_inscription\_date**,** presence: **true**

validates :duration**,** :numericality **=>** **{**:only\_integer **=>** **true}**

validates :hours**,** :numericality **=>** **{**:only\_integer **=>** **true}**

validates :salary**,** :numericality **=>** **{**:only\_integer **=>** **true}**

validates\_format\_of :start\_date**,** :with **=>** /\d{2}\/\d{2}\/\d{4}/

validates\_format\_of :finish\_date**,** :with **=>** /\d{2}\/\d{2}\/\d{4}/

validates\_format\_of :finish\_inscription\_date**,** :with **=>** /\d{2}\/\d{2}\/\d{4}/

Una vez visto los diferentes campos y sus validaciones podemos crear nuestra prueba para los campos con presencia requerida del modelo Offer

describe Offer do

it "is valid with a name, description, perfil\_id,

duration, location and finish\_inscription\_date" do

offer **=** Offer.new**(**

name**:** 'Practicas de PHP',

description**:** 'Oferta de prácticas de PHP',

perfil\_id**:** '3',

duration**:** '300',

finish\_inscription\_date**:** '20/5/2017'**)**

expect**(**offer**)**.to be\_valid

end

it "is invalid without a description" do

offer **=** Offer.new**(**description**:** nil**)**

offer.valid**?**

expect**(**offer.errors**[:**firstname**])**.to include**(**"can't be blank"**)**

end

end

En este ejemplo de prueba lo que hemos hecho ha sido crear un nuevo objeto Offer con datos de ejemplo. Seguido de esto le especificamos el comportamiento que esperamos que tenga con el método “expect” pasándole nuestro objeto Offer recién creado y especificándole que queremos que sea válido con “be\_valid”, ya que lo que nos interesa es verificar que la validación de presencia “presence: true” se cumple.

Una vez construida la prueba podremos ejecutarla y observar si ha ido bien o no, observando aquellos fallos que tengamos en el caso de que la ejecución de la prueba no haya sido exitosa.

Al igual que acabamos de ver, para este proyecto se han hecho diferentes pruebas de esta índole en los modelos más importantes de la aplicación como User, Offer\_Inscription o Message.

6.3 Pruebas de integración y de sistema

En este apartado veremos las pruebas de integración y de sistema, lo que nos permitirá probar la funcionalidad de todo el sistema en conjunto. Gracias a este tipo de pruebas se puede comprobar como es la iteración entre dos o más unidades del sistema, verificando así que los componentes de la aplicación funcionan correctamente en conjunto.

El desarrollo de estas pruebas se ha hecho sobre los casos de uso más importantes de la aplicación:

Tabla 12 Pruebas de integración y sistema

|  |  |
| --- | --- |
| **Caso de uso – Registro de usuario** | |
| **Prueba** | **Resultado Esperado** |
| Añadir un usuario nuevo (no existente) | El sistema registra al nuevo usuario correctamente. |
|  | **Resultado obtenido** |
|  | El sistema registra al usuario correctamente. |
| **Prueba** | **Resultado esperado** |
| Añadir un usuario ya existente en el sistema (mismo email) | El sistema no registra al usuario y se lo notifica. |
|  | **Resultado obtenido** |
|  | Se cumple el resultado esperado, el sistema no registra al usuario y se recibe una notificación. |
| **Prueba** | **Resultado esperado** |
| Añadir un usuario campos requeridos vacíos. | El sistema no registra al usuario y notifica que campos faltan por rellenar. |
|  | **Resultado obtenido** |
|  | Tal y como se esperaba el sistema no añade al usuario y le notifica los fallos. |
| **Caso de uso – Login de usuario** | |
| **Prueba** | **Resultado esperado** |
| Intento de login con un usuario existente en el sistema | El usuario se logea correctamente en el sistema. |
|  | **Resultado obtenido** |
|  | El usuario se logea correctamente en el sistema siempre que introduzca la contraseña asociada a su cuenta, si no, el sistema informará de que las credenciales con las que el usuario intenta acceder son erróneas. |
| **Prueba** | **Resultado esperado** |
| Intento de acceso a una sección no autorizada por el usuario | El usuario es redirigido a la pantalla de inicio y se le notifica que no tiene permisos para acceder a esa sección. |
|  | **Resultado obtenido** |
|  | Efectivamente no se le permite el acceso al usuario y se le notifica que no tiene permisos. |
| **Caso de uso – Recuperar contraseña** | |
| **Prueba** | **Resultado esperado** |
| Recuperación de contraseña usuario registrado. | El sistema permite al usuario cambiar la contraseña. |
|  | **Resultado obtenido** |
|  | Efectivamente el usuario puede cambiar la contraseña de acceso. |
| **Prueba** | **Resultado esperado** |
| Recuperación de contraseña usuario no registrado. | El sistema no envía el email con la URL de modificación de la contraseña debido a que el email introducido por el usuario no está registrado. |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema no envía el email al usuario con la URL y no permite la recuperación de contraseña. |
| **Caso de uso – Modificar información básica de cuenta de usuario** | |
| **Prueba** | **Resultado esperado** |
| Se edita la información del usuario. | El sistema actualiza al usuario en base a la nueva información introducida. |
|  | **Resultado obtenido** |
|  | Efectivamente se actualiza la información de usuario. |
| **Caso de uso – Configurar perfil profesional de usuario** | |
| **Prueba** | **Resultado esperado** |
| Un usuario estudiante añade o edita aquellas competencias profesionales que considere oportunas. | El sistema guarda o actualiza las competencias de forma correcta. |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema guarda o actualiza las competencias. |
| **Prueba** | **Resultado esperado** |
| El usuario estudiante borra aquellas competencias profesionales que considere oportunas. | El sistema borrará aquellas competencias que el usuario haya seleccionado. |
|  | **Resultado obtenido** |
|  | Las competencias se borran correctamente. |
| **Caso de uso – Creación de ofertas** | |
| **Prueba** | **Resultado esperado** |
| Usuario empresa crea una oferta de prácticas | El sistema crea la oferta correctamente |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema crea la oferta sin problemas. |
| **Prueba** | **Resultado esperado** |
| Usuario empresa crea una oferta de prácticas con campos requeridos vacíos | El sistema no permite crear la oferta y le notifica al usuario que campos debe rellenar |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema no permite crear la oferta y notificará al usuario al respecto. |
| **Prueba** | **Resultado esperado** |
| El usuario empresa elimina una oferta de prácticas que haya creado anteriormente. | El sistema elimina la oferta y notificará de esto tanto al usuario empresa como a los usuarios estudiantes que se hayan inscrito en la oferta o estén participando en algunas de las fases del proceso de selección de esta. La notificación a los usuarios estudiantes se hará por medio de mensajes automáticos que mandará el sistema. |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema borra la oferta y notifica tanto a la empresa como a los estudiantes, a estos últimos por medio de un mensaje. |
| **Caso de uso – Inscripción en ofertas** | |
| **Prueba** | **Resultado esperado** |
| El usuario estudiante se inscribe en una oferta de prácticas. | El sistema inscribe al estudiante en la oferta |
|  | **Resultado obtenido** |
|  | Efectivamente el estudiante se escribe en la oferta y se lo notifica |
| **Prueba** | **Resultado esperado** |
| El usuario estudiante se des inscribe de una oferta en la que ya estaba escrito. | El sistema des inscribe al estudiante de la oferta y se lo notifica. |
|  | **Resultado obtenido** |
|  | Efecitvamente el sistema des inscribe al estudiante de la oferta y se lo notifica. |
| **Caso de uso – Proceso de selección** | |
| **Prueba** | **Resultado esperado** |
| Al crear una oferta se crea automáticamente un proceso de selección para esta | El sistema creará un proceso de selección asociado a la oferta de prácticas que se acaba de crear. |
|  | **Resultado obtenido** |
|  | Efectivamente se crea un proceso de selección cuando se crea una nueva oferta. |
| **Prueba** | **Resultado esperado** |
| El usuario empresa elige a aquellos estudiantes de la primera fase del proceso de selección (fase de preselección) para avanzar a la fase final | El sistema avanza de fase a aquellos estudiantes seleccionados por la empresa en la fase de preselección y se lo notifica a estos por medio de un mensaje. El sistema también notificará a aquellos estudiantes no seleccionados por medio de mensajes cuando finalice el plazo de la fase de preselección. |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema avanza de fase a aquellos estudiantes seleccionados y se lo notifica por medio de mensajes. También notifica a los estudiantes no seleccionados. |
| **Prueba** | **Resultado esperado** |
| El usuario empresa elige a aquel estudiante de la fase final del proceso de selección (fase de preselección) que será el ganador de las prácticas. | El sistema selecciona al estudiante ganador y se lo notifica por medio de un mensaje. El sistema también notificará a aquellos estudiantes no seleccionados por medio de mensajes cuando finalice el plazo de la fase final. |
|  | **Resultado obtenido** |
|  | Efectivamente el sistema selecciona al usuario ganador de las prácticas y se lo notifica por medio de un mensaje. También notifica a los estudiantes no seleccionados. |
| **Caso de uso – Envío de mensajes** | |
| **Prueba** | **Resultado esperado** |
| El usuario envía un mensaje a otro. | El mensaje se envía correctamente. |
|  | **Resultado obtenido** |
|  | Efectivamente el mensaje se envía correctamente. |
| **Prueba** | **Resultado esperado** |
| Envío de mensaje con los campos requeridos vacíos. | El sistema no envía el mensaje y se lo notifica al usuario. |
|  | **Resultado obtenido** |
|  | Efectivamente el mensaje no se envía y se notifica al usuario los campos que debe rellenar obligatoriamente. |
| **Caso de uso – Gestión de mensajes** | |
| **Prueba** | **Resultado esperado** |
| El usuario responde a un mensaje | El sistema envía el mensaje de respuesta |
|  | **Resultado obtenido** |
|  | Efectivamente el mensaje se envía correctamente. |

6.4 Pruebas de usabilidad y accesibilidad

A continuación se detallará la evaluación que se ha seguido para los aspectos de usabilidad y accesibilidad de la aplicación web. Se ha decidido seguir una guía heurística basada en los siguientes criterios:

* Generales.
* Identidad e información.
* Lenguaje e información.
* Rotulado.
* Navegación.
* Lay-out de la página.
* Búsqueda.
* Ayuda.
* Accesibilidad.
* Control y retroalimentación.

6.4.1 Criterios generales

Tabla 13 Pruebas de usabilidad y accesibilidad - Criterios generales

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿Cuáles son los objetivos del sitio web? ¿Son concretos y bien definidos? ¿Los contenidos y servicios que ofrece se corresponden con esos objetivos?** Al ser una plataforma pensada para el ámbito profesional esta se ha desarrollado siguiendo unos objetivos claros y concisos, ofreciendo servicios únicamente relacionados con los objetivos previamente establecidos. | **Sí** |
| **¿Tiene una URL correcta, clara y fácil de recordar? ¿Y las URL de sus páginas internas? ¿Son claras y permanentes?** La aplicación no se ha subido a un servidor online, por lo tanto no se ha podido establecer una URL principal, pero las url’s de sus páginas internas son intuitivas, cortas y concisas, siendo fácilmente recordables por el usuario. | **Sí** |
| **¿Muestra de forma precisa y completa qué contenidos o servicios ofrece realmente el sitio web?** La pantalla principal contiene una información breve y concisa de los objetivos principales de la aplicación para los diferentes tipos de usuarios que la utilizarán. | **Sí** |
| **¿La estructura general del sitio web está orientada al usuario?** Dado que la aplicación está pensada para que la utilicen estudiantes y empresas, esta se ha diseñado con una estructura totalmente pensada y orientada al usuario. | **Sí** |
| **¿El look & feel general se corresponde con los objetivos, características, contenidos y servicios del sitio web?** Dado que se trata de una aplicación de ámbito profesional esta se ha diseño con colores oscuros en su mayoría a la vez que claros, intentando transmitir seriedad y evitando colores llamativos y fuera de lugar que puedan confundir al usuario. | **Sí** |
| **¿Es coherente el diseño general del sitio web?** Si lo es, ya que todas las páginas internas de la aplicación siguen una línea de diseño idéntica, incluyendo elementos limpios y con una estructura similar. | **Sí** |
| **¿Es reconocible el diseño general del sitio web?** Dado que el diseño de la aplicación web se ha hecho en su mayoría con colores oscuros, esto proporcionará una clara diferenciación frente a muchas páginas web del mercado, haciendo que el usuario pueda reconocer su diseño más fácilmente. | **Sí** |
| **¿El sitio web se actualiza periódicamente? ¿Indica cuándo se actualiza?** Dado que esta aplicación web ha sido fruto de la realización de un proyecto puramente académico no se ha contemplado el desarrollo de futuras actualizaciones, excluyendo así cualquier tipo de indicación a cerca de estas. | **No** |

6.4.2 Criterios de identidad e información

Tabla 14Pruebas de usabilidad y accesibilidad - Criterios de identificación e información

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿Se muestra claramente la identidad de la empresa-sitio a través de todas las páginas?** Sí, dado que el logo de la aplicación web siempre se mostrará en el menú principal de navegación ubicado en la parte superior de la página. | **Sí** |
| **El Logotipo, ¿es significativo, identificable y suficientemente visible?** El logotipo cuenta con la palabra “intern” que significa becario o alumno en prácticas y la palabra “me” que se refiere a uno mismo, lo cual manda un mensaje significativo a cerca de uno de los principales objetivos de la aplicación, que los estudiantes puedan encontrar prácticas de trabajo. A parte de esto, el logotipo cuenta con una buena calidad, por tanto se puede ser sin problemas en el menú superior de navegación. | **Sí** |
| **El eslogan o tagline, ¿expresa realmente qué es la empresa y qué servicios ofrece?** Dado el slogan pensado para la aplicación “construye tu futuro” podemos afirmar que expresa uno de los principales servicios de la aplicación, y es que los estudiantes vayan construyendo su futuro (laboral) por medio de la realización de prácticas. | **Sí** |
| **¿Se ofrece algún enlace con información sobre la empresa, sitio web, 'webmaster',...?** Como se ha mencionado anteriormente, dado que esta aplicación web ha sido fruto de la realización de un proyecto puramente académico no se ha contemplado la opción de crear algún enlace con información sobre la empresa o sitio web. | **No** |

6.4.3 Criterios de lenguaje y redacción

Tabla 15 Pruebas de usabilidad y accesibilidad - Criterios de lenguaje y redacción

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿El sitio web habla el mismo lenguaje que sus usuarios?** Sí, dado que es una plataforma web pensada para tener estudiantes y empresas del territorio español. Establecer la plataforma en otros idiomas es una posible ampliación que no se debería descartar. | **Sí** |
| **¿Emplea un lenguaje claro y conciso?** Sí, dado que las palabras, oraciones o cualquier otro texto incluido en la aplicación se ha escrito de forma concisa con el fin de explique en el menor número de palabras el mensaje que quiere transmitir | **Sí** |
| **¿Es amigable, familiar y cercano?** Sí, dado que se el usuario tiene muchas interacciones con el sistema, se ha optado por redactar aquellos mensajes o notificaciones que le lleguen al usuario de una forma correcta pero a la vez amigable y cercana. | **Sí** |

6.4.4 Criterios de rotulado

Tabla 16 Pruebas de usabilidad y accesibilidad - Criterios de rotulado

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **Los rótulos, ¿son significativos?** Respecto a los rótulos de los botones o enlaces si se consideran significativos, ya que se han escrito de tal forma que el usuario vea de forma clara y sencilla la acción a realizar en cada uno de ellos. | **Sí** |
| **¿Usa rótulos estándar?** Sí, ya que para el usuario será más fácil navegar por la aplicación web si conoce de ver en otras webs la mayoría de los rótulos. | **Sí** |
| **El título de las páginas, ¿Es correcto? ¿Ha sido planificado?** Sí, cada página o pestaña conllevan un título que describe en su totalidad el contenido de esta. | **Sí** |

6.4.5 Criterios de navegación

Tabla 17 Pruebas de usabilidad y accesibilidad - Criterios de navegación

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿Los enlaces son fácilmente reconocibles como tales? ¿su caracterización indica su estado (visitados, activos,...)?** Sí, ya que se ha optado por un diseño distinto a la hora de incluir enlaces en la aplicación (distinto color, subrayado, etc). | **Sí** |
| **En menús de navegación, ¿Se ha controlado el número de elementos y de términos por elemento para no producir sobrecarga memorística?** Sí, dado que el menú principal de la aplicación solo tiene 3 elementos para cada rol de usuario, evitando así cualquier tipo de sobrecarga con muchos subapartados. | **Sí** |
| **¿Es predecible la respuesta del sistema antes de hacer clic sobre el enlace?** Sí, ya que el enlace expresa la acción a realizar, por lo que el usuario puede hacerse una idea lo que la aplicación le va a mostrar. | **Sí** |
| **¿Se ha controlado que no haya enlaces que no llevan a ningún sitio?** Sí, todos aquellos elementos que se consideran enlaces han sido probados para que cumplan su función y evitar así enlaces que no funcionen. | **Sí** |
| **¿Existen elementos de navegación que orienten al usuario acerca de dónde está y cómo deshacer su navegación?** A pesar de que no se han incluido migas de pan en la aplicación lo que si se ha incluido han sido títulos claros en todas las páginas para que el usuario sepa en todo momento donde se encuentra. | **Sí** |
| **¿Se ha evitado la redundancia de enlaces?** Sí, dado que en las diferentes páginas de la aplicación no se repiten enlaces con la misma funcionalidad. | **Sí** |
| **¿Se ha controlado que no haya páginas "huérfanas"?** Sí, todas las páginas incluidas en la aplicación se han revisado para asegurar que no haya páginas huérfanas accesibles por los usuarios. | **Sí** |

6.4.6 Lay-out de la página

Tabla 18 Pruebas de usabilidad y accesibilidad - Criterios de layout de la página

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿Se aprovechan las zonas de alta jerarquía informativa de la página para contenidos de mayor relevancia? (como por ejemplo la zona central de la página)** Sí, dado que el contenido principal se ha estructurado de tal forma que siempre quede en la zona central de las páginas. | **Sí** |
| **¿Se ha evitado la sobrecarga informativa?** Sí, como se ha explicado antes, cualquier información relativa a los contenidos de la página web se ha minimizado de tal forma que sea breve y clara. | **Sí** |
| **¿Es una interfaz limpia, sin ruido visual?** Sí, dado que no se incluyen más elementos que los requeridos, es decir, no hay imágenes u otros elementos extras sin importancia que distraigan la atención del usuario y que sobrecarguen la página. | **Sí** |
| **¿Existen zonas en "blanco" entre los objetos informativos de la página para poder descansar la vista?** Sí, se ha hecho un diseño de interfaz que incluya márgenes entre los diferentes elementos de tal forma que la información no esté aglutinada y sea más fácil de leer por el usuario. | **Sí** |
| **¿Se ha controlado la longitud de página?** Sí, se ha intentado en todo momento desarrollar las diferentes páginas de la aplicación web con el objetivo de incluir toda la información más importante en el menor espacio posible de cara a que el usuario no tenga que interactuar con páginas de mucha longitud. | **Sí** |

6.4.7 Criterios de búsqueda

Tabla 19 Pruebas de usabilidad y accesibilidad - Criterios de búsqueda

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿La búsqueda se encuentra fácilmente accesible?** Los apartados de búsqueda se corresponden con las búsquedas de estudiantes, empresas y ofertas, los cuales están ubicados al principio de la página correspondiente. | **Sí** |
| **¿Es fácilmente reconocible como tal?** Sí, ya que en los diferentes inputs de la búsqueda se indica en todo momento que se trata de un buscador. | **Sí** |
| **¿Permite la búsqueda avanzada?** Sí, en el sentido de que a la búsqueda se le puede aplicar un filtrado por varios campos. | **Sí** |
| **¿Muestra los resultados de la búsqueda de forma comprensible para el usuario?** Sí | **Sí** |
| **¿La caja de texto es lo suficientemente ancha?** Sí, ya que todos los inputs de texto son lo suficientemente anchos como para albergar más de 7 u 8 palabras. | **Sí** |
| **¿Asiste al usuario en caso de no poder ofrecer resultados para una consultada dada?** Sí, el sistema notificará al usuario cuando no se encuentren resultados para la búsqueda realizada. | **Sí** |

6.4.8 Criterios de ayuda

Tabla 20 Pruebas de usabilidad y accesibilidad - Criterios de ayuda

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **Si posee una sección de Ayuda, ¿Es verdaderamente necesaria?** Sí, ya que al ser una aplicación web que requiere un alto grado de interacción entre el usuario y esta, será necesario explicar el funcionamiento de las funcionalidades a los distintos usuarios | **Sí** |
| **En enlace a la sección de Ayuda, ¿Está colocado en una zona visible y "estándar"?** Sí, ya que está colocado en el menú superior de la aplicación, el cual es visible en todo momento. | **Sí** |

6.4.9 Accesibilidad

Tabla 21 Pruebas de usabilidad y accesibilidad - Criterios de accesibilidad

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿El tamaño de fuente se ha definido de forma relativa, o por lo menos, la fuente es lo suficientemente grande como para no dificultar la legibilidad del texto?** Si, la fuenta y el tamaño de esta se han escogido de forma que el usuario no tenga problemas de legibilidad. | **Sí** |
| **¿El tipo de fuente, efectos tipográficos, ancho de línea y alineación empleados facilitan la lectura?** Sí, ya que se utiliza un ancho de línea que facilitará la lectura al usuario, así como el resultado de la tipografía en las frases u palabras de importancia. | **Sí** |
| **¿Existe un alto contraste entre el color de fuente y el fondo?** Sí, ya que para el usuario será más fácil leer los textos de la página si las letras se distinguen bien del fondo. El fondo negro irá acompañado de una tipografía clara y el fondo blanco ira acompañado de una tipografía oscura. | **Sí** |
| **¿Incluyen las imágenes atributos 'alt' que describan su contenido?** N/A | **-** |
| **¿Es compatible el sitio web con los diferentes navegadores? ¿Se visualiza correctamente con diferentes resoluciones de pantalla?** Sí, ha sido probado para los diferentes navegadores más usados hoy en día. | **Sí** |
| **¿Puede el usuario disfrutar de todos los contenidos del sitio web sin necesidad de tener que descargar e instalar plugins adicionales?** Sí, esta aplicación web no requiere de plugins adicionales para su uso. |  |
| **¿Se ha controlado el peso de la página?** Ya que esta aplicación no cuenta apenas imágenes u otros elementos de peso el peso de la página se reduce al texto que hay en ella, por tanto es ligera. | **Sí** |
| **¿Se puede imprimir la página sin problemas?** Sí | **Sí** |

6.4.10 Criterios de control y retroalimentación

Tabla 22 Pruebas de usabilidad y accesibilidad - Criterios de control y retroalimentación

|  |  |
| --- | --- |
| **Criterios** | **¿Se ha cumplido?** |
| **¿Tiene el usuario todo el control sobre el interfaz?** Sí, ya que no cuenta con banners o pop-ups ajenos al control del usuario. | **Sí** |
| **¿Se informa constantemente al usuario acerca de lo que está pasando?** Sí, cada vez que el usuario interactúe con el sistema, este se lo notificará de una forma u otra. | **Sí** |
| **¿Se informa al usuario de lo que ha pasado?** Sí, el sistema notificará al usuario en todo momento de los eventos que ocurran en la aplicación resultado de la interactuación entre este y el sistema. | **Sí** |
| **Cuando se produce un error, ¿se informa de forma clara y no alarmista al usuario de lo ocurrido y de cómo solucionar el problema?** Sí, se han diseñado mensajes estándar que incluyen el aviso del error así como su solución, de una forma no alarmista. | **Sí** |
| **¿Posee el usuario libertad para actuar?** Sí, ya que no hay ninguna funcionalidad por la que el usuario deba esperar (aunque no quiera). El usuario puede hacer en todo momento lo que quiera dentro de la aplicación. | **Sí** |
| **¿Se ha controlado el tiempo de respuesta?** Debido a la ligereza del lenguaje y entorno utilizado, así como de la rapidez de la base de datos en ejecutar las consultas, esta aplicación genera tiempos de respuestas rápidos en sus peticiones (milisegundos). | **Sí** |

6.5 Pruebas de rendimiento

Una de las características de rendimiento más importantes de esta aplicación web tiene que ver con Javascript, concretamente con la técnica de desarrollo web AJAX (Asynchronous Javascript and XML). Está técnica tiene una gran importancia ya que se ha añadido en la mayoría de funcionalidades de la aplicación, haciendo a esta más ligera de cara a la interacción con el usuario con tiempos de respuesta menores.

La principal característica de Ajax es que permite comunicarse con el servidor (pasándole los datos que se requieran en la petición) y retornar los datos que el servidor proporciona, pero sin necesidad de recargar la página web. Es decir, explicándolo con un ejemplo concreto de nuestra aplicación web, si un usuario crea una oferta de prácticas o envía un formulario para actualizar la información de su cuenta, solo se modificarán aquellos elementos del DOM de la página que nosotros queramos, sin necesidad de volver a recargar la página web y por ente todos sus elementos, proporcionando un rendimiento mucho mayor a la aplicación.

A parte de utilizar técnicas como Ajax para mejorar el rendimiento de a aplicación web, se han llevado a cabo otro tipo de mejoras para aumentar el rendimiento de nuestra aplicación:

* No duplicar código JavaScript (duplicidad de funciones).
* No incluir elementos pesados que ralenticen la aplicación (imágenes grandes, vídeos, etc.)
* Cargar las llamadas a las hojas de estilo CSS al principio de la página y agrupar las llamadas, para que la página cargue los estilos de los elementos lo antes posible.
* Hacer las llamadas a los scripts de Javascript lo más abajo posible en cada página, ya que como la mayoría de funciones actúan sobre los elementos del árbol DOM de la página, estos tendrán que estar cargados previamente para que luego las funciones Javascript los localicen sin problema.

En el caso de tener la aplicación web subida en un servidor online, sería interesante utilizar la herramienta Google PageSpeed Insights (<https://developers.google.com/speed/pagespeed/insights/>) ya que esta se encarga de mostrarnos la velocidad de acceso que tiene nuestra página web así como de analizar sugerir que componentes debemos cambiar para mejorar el rendimiento.

# DOCUMENTO 7: MANUALES

**D. MARTÍNEZ SUÁREZ, Wenceslao**

**TUTOR: Dña. SUAREZ CABAL, María José**

**FECHA: Julio 2017**

Índice de contenidos

[**DOCUMENTO 7: MANUALES** 139](#_Toc486815260)

[7.1 INTRODUCCIÓN 141](#_Toc486815261)

[7.2 LIBRERIAS 141](#_Toc486815262)

[7.2.1 BOOTSTRAP 141](#_Toc486815263)

[7.2.2 BYEBUG 141](#_Toc486815264)

[7.2.3 JQUERY 142](#_Toc486815265)

[7.2.4 NESTED FORMS 142](#_Toc486815266)

[7.2.5 PARANOIA 142](#_Toc486815267)

[7.2.6 POSTGRESQL 142](#_Toc486815268)

[7.2.7 POSTGRESQL SEARCH 142](#_Toc486815269)

[7.2.8 RAILROADY 143](#_Toc486815270)

[7.2.9 RSPEC 143](#_Toc486815271)

[7.2.10 WILL\_PAGINATE 143](#_Toc486815272)

[7.3 MANUAL DE INSTALACIÓN Y EJECUCIÓN 143](#_Toc486815273)

[7.3.1 BASE DE DATOS 143](#_Toc486815274)

[7.3.2 RUBY VERSION MANAGER 144](#_Toc486815275)

[7.3.3 EJECUCIÓN 145](#_Toc486815276)

7.1 Introducción

En el apartado XXX del documento XXX presentamos las tecnologías y herramientas utilizadas en el desarrollo de esta aplicación web con sus versiones. A continuación y con el objetivo de incluir aquella información necesaria para la instalación y la utilización del sistema, se describirán las librerías utilizadas en él y se detallarán los pasos a seguir para la correcta instalación en un equipo.

7.2 Librerías

A continuación se detallarán aquellas librerías, conocidas cono gemas en Ruby on Rails, más importantes que han sido utilizadas para el desarrollo de los diferentes módulos de la aplicación. Estas gemas se declaran en el fichero Gemfile, ubicado en la raíz del proyecto. Cada vez que queramos agregar una gema al proyecto tendremos que agregarlas al fichero Gemfile, y en el caso de desinstalarlas tendremos que borrarlas de ese fichero. Para instalar aquellas gemas incluidas en el Gemfile debemos ejecutar la siguiente orden en la consola de comandos:

$ bundle install

7.2.1 Bootstrap

Esta gema se ha instalado debido a la integración del framework de diseño Bootstrap. Gracias a esta gema y como se ha dicho en apartados anterior, se han podido integrar las diferentes hojas de estilo CSS y el diseño responsive con el que cuenta.

7.2.2 Byebug

Esta gema ha jugado un papel importante en el desarrollo de este trabajo fin de grado, y es que esta actúa como un depurador, controlando la ejecución del sistema en todo momento y proporcionando puntos de interrupción a ubicar donde el desarrollador desee, con el objetivo de controlar la traza de ejecución entre las diferentes llamadas del sistema.

7.2.3 jQuery

Esta gema se ha utilizado para importar la biblioteca de Javascript Jquery, la cual permite interactuar con los elementos DOM de una página web, el manejo de eventos o agregar la técnica AJAX. Actualmente es la biblioteca más usada de Javascript en la actualidad.

7.2.4 Nested forms

Esta gema se ha incluido en el sistema para la utilización de formularios anidados, los cuales se trabajan con modelos relacionados y generan una mejor experiencia de usuario, ya que se podrán integrar formularios de un modelo específico dentro del formulario otro modelo diferente en el caso de estar relacionados.

7.2.5 Paranoia

Esta gema permite almacenar aquellos objetos eliminados del sistema haciéndolos invisibles, de tal forma que si se produce el borrado de un objeto de forma inintencionada, este objeto seguirá en la base de datos con toda su información.

7.2.6 PostgreSQL

Esta gema sirve para establecer la conexión entre la aplicación web y nuesta base de datos PostgreSQL.

7.2.7 PostgreSQL Search

Esta gema se ha implementado para la creación de los buscadores en las páginas de Estudiantes, Empresas y Ofertas de la aplicación. Esta gema hace las búsquedas dentro de la base de datos PostgreSQL más fáciles y sencillas.

7.2.8 RailRoady

Esta gema se ha utilizado para la generación de diagramas UML de los modelos y controladores de la aplicación.

7.2.9 Rspec

Como hemos explicado detalladamente EN EL APARTADO DE PRUEBAS UNITARIAS, esta gema se ha utlizado para la realización de pruebas unitarias en los diferentes modelos más importantes de la aplicación.

7.2.10 Will\_paginate

Esta gema proporciona una API para desarrollar la paginación de objetos en nuestra aplicación web, tanto en las vistas como en los controladores.

7.3 Manual de instalación y ejecución

En primer lugar y como mencionamos en el apartado XX, el desarrollo de esta aplicación web se ha realizado sobre una distribución Linux, concretamente en Ubuntu 14.04 LTS, por tanto la instalación del entorno de trabajo se desarrollará sobre esta distribución.

7.3.1 Base de datos

En primer lugar instalaremos el sistema de bases de datos PostgreSQL y la herramienta pgAdmin3, con la que podremos gestionar gráficamente sus bases de datos. En la consola de comandos como super-usuario debemos introducir las siguientes líneas y ejecutarlas:

$ sudo apt-get update

$ sudo apt-get install postgresql postgresql-contrib libpq-dev

Una vez instalado PostgreSQL, arrancamos en su consola de comandos y configuramos la contraseña para nuestro usuario de postgres, en este caso llamado ‘postgresql’:

$ sudo -u postgres psql postgres

postgres**=**# \password postgres

Enter new password:

Enter it again:   
postgres**=**# \q

Una vez configurado nuestro usuario de postgres, instalamos pgAdmin3:

$ sudo apt-get install pgadmin3

7.3.2 Ruby Version Manager

Una vez completada la instalación de PostgreSQL y pgAdmin3, procederemos a instalar el framework Ruby on Rails. Una de las ventajas por las que se caracteriza la instalación de este framework es la rapidez con que se puede hacer, ya que utilizando ‘Ruby Version Manager (RVM)’ podremos realizar la instalación de Ruby on rails de manera sencilla. RVM actúa como un gestor de versiones que te permite instalar y mantener varias versiones de ruby a la vez. A continuación se detallan los pasos:

1. Primer instaamos los paquetes del sistema operativo requeridos por RVM:

$ sudo apt-get install -y git-core subversion

2. En segundo lugar instalaremos la firma requerida por RVM y procederemos a la instalación principal. La segunda orden que veremos a continuación instala las últimas versiones estables de rvm, ruby y Rails.

$ gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3

$ \curl -sSL https://get.rvm.io **|** bash -s stable --rails --ruby

3. Por último, la versión de Ruby que se ha utilizado en el desarrollo de este proyecto ha sido la versión ‘2.3.3’. La instalaremos de la siguiente forma:

$ rvm install 2.3.3

4. Una vez instalada la versión de Ruby con la que trabajaremos procederemos a instalar Rails dentro de la carpeta donde esté ubicado el trabajo. Para ello haremos uso de la gema bundler, instalándola en primer lugar.

$ sudo gem install bundler  
$ bundle install

La segunda orden ‘bundle install’ instalará Rails, así como todas sus gemas declaradas en el fichero Gemfile y sus dependencias.

7.3.3 Ejecución

Una vez hemos instalado el entorno de forma correcta, procederemos a arrancar la aplicación. En primer lugar debemos configurar la base de datos y luego crearla.

La configuración de la base de datos para el entorno de desarrollo está en el archivo ‘database.yml’ dentro de la carpeta ‘config’ ubicada en el directorio raíz de la aplicación. La configuración debe seguir estas declaraciones:

**development**:

**adapter**: postgresql

**encoding**: unicode

**database**: tfg\_v1\_development

**pool**: 5

**host**: localhost

**username**: postgres  
 **password**: hubb16

Una vez asegurados de que tenemos correctamente configurado el archivo ‘database.yml’ debemos abrir la consola de comandos y ubicados dentro de la carpeta donde esté el proyecto ejecutar las siguientes órdenes:

$ rake db:create  
$ rake db:seeds

La primera orden creará nuestra base de datos, mientras que la segunda la poblará con la información de prueba que he añadido en el fichero ‘seeds.rb’ ubicado en la carpeta ‘/db’.

Una vez instalados el entorno y la base de datos, podremos arrancar nuestra aplicación. El puerto por defecto donde Rails lanza nuestra aplicación es el puerto 3000, por lo tanto en nuestro navegador accederemos a nuestra aplicación escribiendo ‘localhost:3000’, siempre y cuando tengamos arrancado nuestro servidor, ejecutando para ello en la consola de comandos la orden ‘rails s’.