UNIVERSIDAD
REY JUAN CARLOS

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE LA TELECOMUNICACIÓN

Curso Académico 2016/2017

Trabajo de Fin de Grado

WELPE: DESARROLLO DE UNA APLICACIÓN WEB PARA LA GESTIÓN DE EVENTOS

Autor : Itziar Polo Martínez

Tutor : Dr. Gregorio Robles

II

Dedicado a

quien considero mi familia.

II

Agradecimientos

Quiero agradecérselo a Olatz, por todos esos días y sobre todo noches que me ha acompañado, proporcionándome ánimos para seguir adelante y no rendirme nunca.

A mis padres que sin su apoyo no estaría donde estoy. GRACIAS.

También quiero agradecérselo a Carlos, mi apoyo día a día, mi compañero en esta aventura a la que llamamos vida. Gracias por confiar en mí.

A mis amigos, en especial a Marta y Patxi que pese a todas las dificultades vividas siguen estando ahí.

A mis compañeros que más que compañeros son amigos. Junto a ellos he vivido momentos buenos y no tan buenos, pero siempre sacando algo positivo, siempre aprendiendo algo.

Por último quiero agradecérselo a mis profesores que sin sus conocimientos esto no hubiera sido posible.

A todos vosotros, por si alguna vez no os lo he dicho lo suficientemente alto: GRACIAS

Resumen

Este trabajo desarrolla una herramienta para la creación y gestión de los diversos mensajes que los profesores y alumnos quieren compartir. Pretende, por tanto, mejorar la gestión y promocionar los distintos eventos relacionados con la universidad o mejor dicho, relacionados con la vida universitaria, como por ejemplo ofertas de trabajo y prácticas en empresas, cursos, becas, concursos, seminarios, etc.

FIXME: comentar algo sobre las tecnologías utilizadas

Summary

This project develops a tool for the creation and management of the various events that the teachers and students want to share. It aim, therefore, to improve the management and promotion of the different events related to the university, or better said, related to university life, such as job offers and internship, courses, competitions, seminars, etc.

Índice general

Índice de figuras

Capítulo 1

Introducción

En este primer capítulo incluye una sección de presentación del presente trabajo; también se aborda un pequeño epígrafe sobre las motivaciones que han propiciado la elaboración del mismo, así como se concluirá con una reseña a la estructura que sigue este proyecto.

1.1. Presentación

Este documento describe el trabajo realizado en el trabajo de fin de carrera de grado en ingeniería en tecnologías de la telecomunicación. El proyecto consiste en la creación de una aplicación Web para la creación y la gestión de eventos. Este proyecto pretende dar un servicio actualmente no existente o si existiera algo, no usable actualmente en el momento de elaboración de este TFG.

Por lo tanto, con esta nueva aplicación conseguiremos mejorar la gestión y promocionar los distintos eventos relacionados con la universidad como por ejemplo ofertas de trabajo y prácticas en empresas, cursos, becas, concursos, seminarios, etc.

El proyecto ofrecerá un servicio deslocalizado con el que se podrá acceder desde cualquier sitio y en cualquier momento.

En cuanto a los usuarios, la página tiene varios tipos. Por un lado, están los usuarios no registrados que únicamente pueden ver los distintos eventos. Por otro, están los usuarios registrados que pueden acceder a una mayor funcionalidad dentro de la aplicación. Entre estos últimos se encuentra el usuario alumno, el usuario profesor y el usuario administrador. Los usuarios alumnos, profesor y administrador pueden realizar acciones tales como añadir y comentar un evento

además de participar en el foro. Además los usuarios del grupo de profesor y administrador podrán crear los distintos eventos y asignar los eventos denominados como actividades a otros usuarios.

Para crear la aplicación se ha utilizado el gestor de contenidos Mezzanine y como base de datos SQLite3. Se ha escogido este tipo de herramientas por su fácil acceso y disponibilidad así como por su sencillez tanto de instalación como de uso para usuarios no técnicos. Para su despliegue s ha utilizado Amazon Web Services (aws). Se ha escogido esta herramienta por ser muy usada y ofrecer todos sus servicios gratuitos durante 1 año. Se ha escogido Ubuntu Server 14.04 LTS como sistema operativo del servidor.

Para finalizar, quiero señalar un par de puntos. Para mí, ya que el TFG supone el último paso para obtener un premio tan merecido como ansiado, creo que cada TFG debe llevar una parte de cada uno. Por este motivo, tanto la elección de colores como el logotipo y el nombre de la presente aplicación tienen un significado especial.

Los colores elegidos tanto el azul como ese rosa en particular son colores que me atraen bastante. Y dado que me gustan no podía no incluirlos como colores principales. Respecto al nombre de la aplicación, debo ser sincera y decir que me costó encontrar un nombre que me gustara y que pudiera tener un significado especial para mí. Había muchos pretendientes pero ninguno me llamaba la atención, hasta que encontré esa palabra alemana que además de sonar bien significaba algo para mí. Lo he llamado *Welpe*, que para el lector que no esté familiarizado con el alemán viene a significar cachorro de perro, con lo que era todo un homenaje a mi perra. El logotipo sería mucho más sencillo con un nombre elegido. Si el proyecto se llamaba Welpe, la mejor idea sería una huella de perro.

1.2. Motivación Personal

La motivación personal de este proyecto residen en que ya que no existe ninguna herramienta usada por la universidad, todos los eventos llegan mediante un simple anuncio en el foro de la universidad y para el registro en alguna de las actividades, ésta se realiza mediante el envío de un correo electrónico o rellenando un simple ”google forms”, además si estamos realizando una actividad para que los asistentes puedan tener acceso al material de estudio es prácticamente imposible sin copiar correo a correo o escribir un anuncio en el foro donde los demás usuarios

no registrados en esa actividad lo verían.

1.3. Estructura de la memoria

Esta sección trata de detallar la estructura que se sigue a lo largo del proyecto para facilitar su orden y mejor comprensión.

- Capítulo I. Introducción: se realiza una descripción general del proyecto mostrando la motivación para el desarrollo del mismo y finalizando con una breve explicación de la estructura del mismo.
- Capítulo II. Objetivos: se muestran cuál han sido los objetivos generales y específicos que se pretende lograr con la ejecución de este trabajo.
- Capítulo III. Estado del arte: describe las distintas tecnologías utilizadas en la realización de la aplicación web.
- Capítulo IV. Diseño e implementación: se describe la aplicación diseñada además se describe su estructura y su funcionamiento.
- Capítulo V. Resultados: se analizan los resultados obtenidos.
- Capítulo VI. Conclusiones: se comentan las conclusiones finales tras la realización del trabajo y se comentan posibles líneas de desarrollo futuras.

Capítulo 2

Objetivos

En esta sección se pretende recoger todos los objetivos que han motivado la realización y desarrollo de esta herramienta.

2.1. Objetivo general

El objetivo de este trabajo se basa en el desarrollo de una aplicación web para la creación y gestión de toda clase de eventos.

2.2. Objetivos específicos

Los principales objetivos que han guiado el desarrollo han sido:

- Aplicación universal: este es uno de los objetivos más importantes; desarrollar una aplicación que pueda utilizarse desde cualquier plataforma, para que todos los usuarios puedan acceder desde sus dispositivos.
- Sencillez en su uso: la aplicación será utilizada por una gran variedad de personas, algunas de las cuales pueden no estar muy familiarizadas con las nuevas tecnologías, por lo que debe ser fácil de utilizar. Tiene que ser un diseño sencillo e intuitivo que sea atractivo y que incite el uso de la aplicación web.
- Gestionar y promocionar mejor las actividades para que los alumnos crezcan tanto personal como profesionalmente. También se conseguirá que los alumnos tengan algo más

de voz sobre las actividades que desean que se realicen, esto se consigue mediante el apartado de propuestas.

- Web colaborativa: toda la información contenida en esta aplicación web deberá ser proporcionada por los usuarios de la misma tanto profesores creando los distintos eventos como los alumnos creando sus propuestas.

- Aportar un elemento social: debido al gran interés que despiertan las redes sociales, es una buena idea integrar alguna función social para atraer a los usuarios, es importante agradarles para conseguir un mejor despliegue; muchos proyectos acaban fracasando por no tener un buen apoyo social inicial, produciéndose un reclamo por parte de los usuarios para volver al sistema tradicional.

- Uso de tecnologías web avanzadas: en la actualidad el desarrollo web está creciendo a pasos agigantados. En este trabajo se pretenden utilizar tecnologías web muy utilizadas como lo son HTML5, CSS3 y Bootstrap y aprovechar sus funcionalidades.

Capítulo 3

Estado del arte

En este capítulo se describen las tecnologías utilizadas para la realización de la aplicación.

3.1. Python

Python fue desarrollado a finales de los 80 por Guido van Rossum en los Países Bajos, como sucesor del lenguaje de programación ABC, capaz de maneja excepciones e interactuar con el sistema operativo Amoeba.

El nombre de Python proviene de la afición de van Rossum, a los humoristas británicos Monty Python.

Desde el principio se buscó que fuera un lenguaje divertido a la hora de utilizarlo y es muy común utilizar variables que hacen referencia a sketchs de los Monty Python en lugar de utilizar las variables tradicionales.

Es un lenguaje que destaca por ser rápido, potente, comportarse bien con otros lenguajes, ser compatible con los diversos sistemas operativos más utilizados, fácil de aprender y de código abierto. También podemos destacar que, Python es un lenguaje de alto nivel que permite realizar operaciones complejas en pocas líneas de código, lo que facilita escribir programas en menos tiempo, y dedicarle más tiempo al diseño y a la depuración del código.

Existen tres versiones principalmente, con 1.6, 2.7 y 3.5 como las últimas actualizaciones.

Es destacable indicar que las versiones 2.7 y 3.4 son incompatibles entre sí, ya que existen un gran número de diferencias entre ellas.

El desarrollo y promoción se realiza a través de la organización Python Software Founda-

tion, destacando su carácter Open Source.

Dado que el lenguaje es interpretado, no es necesario que se realice compilación. Cabe destacar también, que es un lenguaje dinámicamente tipado con lo que puede indicarse el tipo en cualquier momento, no es necesario declararlos en un método; además de ser dinámicamente tipado es un lenguaje fuertemente tipado, con lo que una vez declarada una variable con un tipo, esta no puede realizar una violación de tipos de datos, sino que previamente deberá ser convertida. A parte de lo mencionado anteriormente, cabe destacar que Python es "case sensitive", es decir, diferencia entre mayúsculas y minúsculas.

Python es un lenguaje de programación multiparadigma, es decir, permite varios estilos de programación: programación orientada a objetos, programación imperativa y programación funcional.

Si se necesita crear un nuevo módulo para integrarlo se puede realizar fácilmente en C o C++.

Por último, resaltar que, los usuarios de Python se refieren a menudo a la Filosofía Python que es bastante análoga a la filosofía de Unix. El código que sigue los principios de Python de legibilidad y transparencia se dice que es "pythonic". Contrariamente, es bautizado como "unpythonic". Estos principios fueron famosamente descritos por el desarrollador de Python, Tim Peters, en The Zen of Python:

Beautiful is better than ugly. Explicit is better than implicit Simple is better than complex. Complex is better than complicated. Flat is better than nested. Sparse is better than dense. Readability counts. Special cases aren't special enough to break the rules. Although practicality beats purity. Errors should never pass silently. Unless explicitly silenced. In the face of ambiguity, refuse the temptation to guess. There should be one– and preferably only one –obvious way to do it. Although that way may not be obvious at first unless you're Dutch. Now is better than never. Although never is often better than *right* now. If the implementation is hard to explain, it's a bad idea. If the implementation is easy to explain, it may be a good idea. Namespaces are one honking great idea – let's do more of those!

3.2. Django

Django es un framework de desarrollo web Open Source, escrito en Python. Fue desarrollado inicialmente para gestionar páginas de noticias de la World Company de Lawrence, Kansas, pero en 2005 fue liberada bajo la licencia BSD.

El nombre de Django proviene del guitarrista de Jazz gitano Django Reinhardt.

Desde 2008, Django Software Foundation se hizo cargo de desarrollarlo y promocionarlo.

Su filosofía principal es facilitar la creación de sitios web complejos, facilitar la reutilización de elementos (creando módulos independientes), disminuir la cantidad de código, dar prioridad a lo explícito frente a lo implícito (cualidad heredada de Python, como vimos anteriormente Explicit is better than implicit), ser consistente en todos los niveles y evitar repetir código o información, siento una de sus principales citas "Don't repeat yourself (DRY)"

Se basa en el paradigma Modelo-Vista-Controlador. Consiste en separar los datos y la lógica de negocio de una aplicación (modelo) de la interfaz de usuario (vista) y del módulo encargado de gestionar las comunicaciones (controlador). Aunque el controlador es llamado vista y la vistas template.

Para su compresión, Django se puede dividir en 5 grandes bloques:

- API de la base de datos: Django soporta abstracciones para interactuar con la base de datos de forma muy sencilla. Sus principales objetivos consisten en lograr eficiencias en el acceso a la base de datos, intentando reducir el número de accesos a la misma.

- Models o modelos: es la definición de los elementos que se almacenan en la base de datos, en general cada modelo equivale a una tabla en la base de datos. Los modelos deberían incluir toda la lógica sobre ellos mismos, desde el tipo de datos que almacena hasta los campos para ser interpretados por los humanos, para una mejor comprensión de un determinado modelo y una mayor organización.

- URLs: es la interfaz entre el cliente y el servidor. Su filosofía defiende desemparejar las URLs de las funciones para una mayor reutilización, además deben ser flexibles.

- Views o vistas: son funciones Python que recibe una petición y devuelven una respuesta. Esta respuesta puede ser cualquier cosa. La vista contiene la lógica para devolver esa

respuesta. Las vistas deben tener acceso a un objeto petición que contenga la información de solicitud (siempre se recibe este objeto como parámetro).

- Templates o plantillas: presenta un sistema para renderizar o generar dinámicamente la respuesta a una petición a partir de unas plantillas previamente creadas y un contexto, que puede variar dependiendo de la petición de la URL, de la información disponible en la base de datos o del usuario que realice la petición. Además busca evitar la redundancia, basándose en que la mayor parte de las páginas web dinámicas comparten gran parte del diseño.

Django tiene una gran comunidad de usuarios que defienden al igual que Python las buenas prácticas de programación y que el código sea legible. Por último una herramienta muy útil que ofrece Django es la API Rest Framework. Se trata de una herramienta muy potente y flexible que facilita mucho el desarrollo del backend , ya que cuenta con acceso rápido a los objetos de la base de datos.

3.3. Mezzanine

Un CMS (Content Management System) es una colección de scripts (o archivos de procesamiento por lotes) que permiten crear una estructura de soporte para la creación y administración de contenidos de una manera ágil e intuitiva. Además, separan el contenido de la presentación del sitio.

Se trata de herramientas que permiten crear y mantener un Web con facilidad, encargándose de los trabajos más tediosos que hasta ahora ocupaban el tiempo de los administradores de las Webs.

Los CMS proporcionan un entorno que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.

Mezzanine es una plataforma de gestión de contenidos de gran alcance, consistente y flexible. Construida bajo Django, Mezzanine provee una arquitectura altamente extensible y simple. Mezzanine posee una licencia BSD (al igual que Django) y apoyo de una comunidad diversa y activa.

En algunos aspectos, Mezzanine se asemeja mucho a Wordpress, proporciona una interfaz

intuitiva para el manejo de páginas, blogs, datos de formulario, productos de tienda y otros tipos de contenido. Pero también es diferente. Mezzanine proporciona mucha funcionalidad por defecto, como por ejemplo los blogs que es una de las aplicaciones que se pueden instalar al crear una nueva aplicación Mezzanine. Este enfoque proporciona una plataforma más integrada y eficiente.

Algunas de sus principales características son:

- Navegación semántica
- Integración con Bootstrap, disqus, bit.ly o Google Analytics
- Formularios HTML5 que permiten drag&drop y edición inline
- Widget configurables
- Etc.

3.4. HTML5

HTLM5 es la quinta versión del lenguaje de marcado HTML.

Con HTML5 los navegadores web más importantes (Firefox, Chrome, Safari e Internet Explorer) pueden saber cómo mostrar una determinada página web, saber dónde están sus elementos e incluso saber dónde colocar el contenido. La diferencia principal entre HTML5 con las versiones anteriores de HTML es el nivel de sofisticación del código.

Esta nueva versión de HTML, proporciona la visualización de contenido multimedia incluso sin estar conectado a la red, evitando el uso de Flash. Incluye nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos.

HTML (HyperText Markup Language). Es un lenguaje de marcado, utilizado para la elaboración de páginas web.

Los documentos HTML comienzan con una declaración para indicar que versión de HTML se está utilizando. El lenguaje está formado por elementos que indican el tipo de contenido que presentan entre su apertura y su cierre, pudiendo anidarse unos elementos dentro de otros. El primer elemento, escrito después de la declaración, es la etiqueta html que se anida directamente con una etiqueta head y una etiqueta body.

- Elemento HEAD: contiene principalmente información general acerca de la página como puede ser el título, la descripción, las keywords, etc. Además suele incluir las referencias a archivos externos, como las hojas de estilo (archivos CSS) y scripts (archivos o código JavaScript), que son utilizados en la propia página. Aunque estos dos elementos (CSS y JS) se pueden incluir directamente, es preferible que se incluyan en documentos externos y limitarse a poner una simple referencia en el HTML.

- Elemento BODY: contiene el contenido propio de la página. Generalmente, cada elemento se indica con una etiqueta de inicio y una etiqueta de fin, aunque hay elementos que se pueden abrir y cerrar en una simple etiqueta, pudiendo incluso contener atributos. Los atributos están formados por par nombre-valor, aunque algunos pueden carecer de valor, y se utilizan para definir características de ese elemento y/o para identificarlos (mediante un atributo id que será único o un atributo class que puede ser compartido por varios elementos).

3.5. CSS3

Las hojas de estilo en cascada, CSS (Cascade Style Sheets) es el lenguaje utilizado para describir la presentación de documentos HTML o XML, aunque también es utilizado para modificar la interfaz de usuario. Aparte de aplicar estilos visuales, es capaz de aplicar estilos no visuales, como las hojas de estilo auditivas.

Para ser incluido en un archivo HTML basta con incluir la etiqueta style indicando el valor de estilo de dicho elemento, también se puede utilizar añadiendo el atributo style dentro de un elemento. La mejor opción de todas es incluir un link a un archivo con extensión css, en la cabecera del documento HTML.

En 1995, el W3C apostó por desarrollar el estándar CSS, llegando a 1996 donde se publicaría la primera versión.

CSS3 es la tercera versión de CSS, ofrece unas opciones verdaderamente importantes que cubren las necesidades del diseño web actual.

CSS3 está dividido en diferentes archivos, llamados módulos. Cada módulo añade nuevas funcionalidades a las definidas por CSS2.

3.6. JavaScript

JavaScript, o simplemente JS, es un lenguaje de programación interpretado que se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Usado principalmente en el desarrollo de aplicaciones web en el lado del cliente, aunque también es utilizado en el lado del servidor. Su uso en aplicaciones externas a la web es también significativo.

Se empezó a desarrollar a principios de los 90, momento en el que los desarrolladores web necesitaban servir sitios web donde los usuarios pudieran interactuar con ellos, en ese momento las páginas web eran estáticas (HTML junto con CSS). Al estar al lado del cliente, no requiere compilación y es el navegador encargado de interpretar el código.

A pesar del nombre, no es un lenguaje similar a Java, ni tampoco tiene el mismo propósito. Su nombre proviene de la popularidad que, en el momento del desarrollo de JS, tenía Java. Surgieron varias versiones similares a JS, hasta que se estandarizó como ECMAScript, para evitar incompatibilidades en los navegadores.

3.7. AJAX

AJAX (Asynchronous JavaScript And XML) es una tecnología que permite enviar y recibir información desde el servidor al cliente, permitiendo desarrollar aplicaciones web más atractivas.

Este envío de información se realiza a través de una comunicación asíncrona con el servidor, siendo posible realizar cambios en la página sin que los clientes la recarguen.

AJAX no es una tecnología basada únicamente en JS, sino que también se basa en HTM y CSS para mostrar el contenido.

Esta tecnología permite el diseño de webs más eficientes al poder solicitar solo la información requerida, aprovechando los datos cargados previamente; a parte, permite crear aplicaciones más fluidas y dinámicas de cara al usuario final.

3.8. jQuery

jQuery es la biblioteca de JavaScript más utilizada.

jQuery es una biblioteca creada por John Resig. Fue presentada el 14 de enero de 2006 en el BarCamp NYC. Es de software libre y posee un doble licenciamiento bajo la licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privados

Esta biblioteca facilita el acceso y la manipulación del árbol DOM, el manejo de eventos y la modificación de estilos (CSS), además de desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

3.9. Bootstrap

Bootstrap es un framework que permite al desarrollador despreocuparse en gran medida del diseño y apariencia al realizar una aplicación web. Está basado en HTML y CSS separando por completo dichos lenguajes en documentos distintos, lo que permite que una plantilla pueda ser modificada fácilmente por el desarrollador. Las plantillas generalmente también incluyen archivos JavaScript para aportar a la página un diseño más atractivo. Fue desarrollado inicialmente por Mark Otto y Jacbod Thornton de Twitter para fomentar la consistencia entre las herramientas internas. En agosto del 2011 Twitter liberó Bootstrap como código abierto. En febrero del 2012, se convirtió en el proyecto de desarrollo más popular de GitHub.

3.10. JSON

JavaScript Object Notation (JSON) es un formato ligero para el intercambio de datos. Es fácil de leer y escribir para los humanos, además también es fácil de generar y parsear por los ordenadores.

Es un formato muy utilizado en la actualidad para enviar información desde un servidor a un cliente ante una petición AJAX, siendo dicha información interpretada por código JavaScript en el cliente, con lo que permite trabajar con estos datos como si se tratase de cualquier otra variable.

3.11. Amazon Web Services intr(AWS)

Amazon Web Services (AWS) es el conjunto de servicios web (computación en la nube) que forman una plataforma de Cloud Computing ofrecida por Amazon.com y es uno de los proveedores de la computación en nube más importantes de todo el planeta.

Todos los servicios que se ofrecen en Amazon Web se clasifican en distintos grupos, los más importantes serán almacenamiento, bases de datos, entrega de contenido, mensajería y procesamiento de datos. Muchas son las aplicaciones que utilizan los servicios de Amazon Web, algunas tan populares como Dropbox, Foursquare o HootSuite entre otras. Además, AWS compite directamente con otros servicios tan importantes como Microsoft Azure y Google Cloud Platform.

De todos los servicios disponibles en AWS se ha escogido EC2.

Amazon Elastic Compute Cloud (Amazon EC2) es el servicio web de Amazon que ofrece en la nube capacidad informática de tamaño modificable. La función de este servicio reside en facilitar a los desarrolladores recursos informáticos escalables y basados en la Web, pagando por su uso en cada instante. La interfaz que tiene este servicio es tan sencilla que permite obtener y configurar la capacidad de manera simple teniendo un control completo sobre los recursos informáticos del cliente.

3.12. Apache

Apache es el servidor Web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Es desarrollado y mantenido por una comunidad de desarrolladores de Apache Software Foundation.

En 1995 comenzó su desarrollo basándose en la aplicación NCSA HTTPd. Era un servidor web desarrollado en el National Center for Supercomputing Applications y que fue cancelado en 1998. Con el paso de los años el código ha sido reescrito hasta ser substituido por lo que conocemos hoy en día.

Aunque no posee interfaz gráfica, su configuración es bastante sencilla y al ser tan popular es muy fácil encontrar información sobre cómo realizar dichas configuraciones.

Otra gran característica de este servidor es que es modular. Consta de un core y diversos módulos que permiten ampliar fácilmente las capacidades del servidor. Existe una gran lista de módulos que podemos instalar en el servidor Apache o incluso, al ser de código abierto, si se poseen los conocimientos necesarios se pueden programar módulos nuevos. De entre todos los módulos existentes hay que destacar los módulos para el uso de SSL, PHP, Perl y Rewrite, además de un módulo para Python.

La mayor parte de la configuración se realiza en el fichero apache2.conf (Ubuntu) o httpd.conf (otros). Cualquier cambio en este archivo requiere reiniciar el servidor, o forzar la lectura de los archivos de configuración nuevamente

También posee bases de datos de autenticación y negociado de contenido y permite personalizar la respuesta ante posibles errores que se puedan dar en el servidor. Se puede configurar para que ejecute scripts cuando ocurra un error en concreto. Permite con mucha facilidad la creación y gestión de logs para tener un mayor control sobre lo que sucede en el servidor.

Su nombre se debe a que alguien quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y “civilizasen” el paisaje que habían creado los primeros ingenieros de Internet. Además Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. En inglés, *a patchy server* (un servidor “parcheado”) suena igual que Apache Server.

Capítulo 4

Diseño e implementación

En este capítulo se realiza una descripción detallada sobre el diseño y la implementación del presente Trabajo de Fin de Grado.

La arquitectura de la aplicación esta diferenciada en el lado del cliente y en el lado del servidor aunque interactúan entre ellos. El cliente está desarrollado con las tecnologías HTML5, CSS3, JavaScript, BootStrap, jQuery, mientras que el servidor se desarrolla en Python, Django y Mezzanine.

4.1. Arquitectura general

FIXME: estaría bien tener un esquema de la arquitectura general en un diagrama.

El proyecto consta de múltiples ficheros y directorios. Aunque se haya utilizado un CMS, la arquitectura sigue siendo la misma que la utilizada en Django. Cada una de las distintas aplicaciones que contiene el presente proyecto mantiene la siguiente estructura:

- Urls.py: es la interfaz entre el cliente y el servidor. Maneja las peticiones HTTP recibidas y las envía a la función contenida en el *views* correspondiente.
- Views.py: recibe la información de la petición HTTP y devuelve una respuesta. En el presente proyecto sólo se reciben peticiones GET y peticiones POST.
- Models.py: determina la estructura de datos.
- Templates: contiene las plantillas HTML que definen la interfaz de usuario

- Static: contiene los ficheros estáticos CSS y JS, además de contener las imágenes utilizadas.

- Utils: contiene funciones llamadas por el *views*. Cada llamada recibe unos parámetros, los procesa y devuelve un resultado fruto de ese procesamiento.

Como se ha comentado anteriormente, cuando el servidor recibe una petición HTTP por parte del cliente, dicha petición es manejada desde el fichero principal urls.py. Tras encontrar la URL en dicho fichero continúa con el siguiente fichero urls.py (que será el de la aplicación pedida) a continuación se hace ejecutar la función correspondiente del fichero views.py. La función trata la petición GET o POST y responderá con una petición HTTP que devolverá la plantilla HTML (con todos los demás ficheros estáticos) correspondiente al cliente.

4.2. Diseño e implementación del servidor

4.2.1. Modelos de la base de datos

La aplicación requiere una base de datos donde se guardará toda la información que los usuarios introduzcan en la aplicación web. La estructura de la base de datos se almacena, como hemos comentado antes, en el archivo models.py (como hemos dicho anteriormente, cada aplicación contiene su propio archivo models.py). Django generará las tablas pertinentes dentro de la base de datos. La base de datos utilizada es SQLite3 que es un tipo de base de datos relacional.

La ventaja que tenemos al usar un Django es que la conexión con los sistemas de la base de datos es transparente para el usuario.

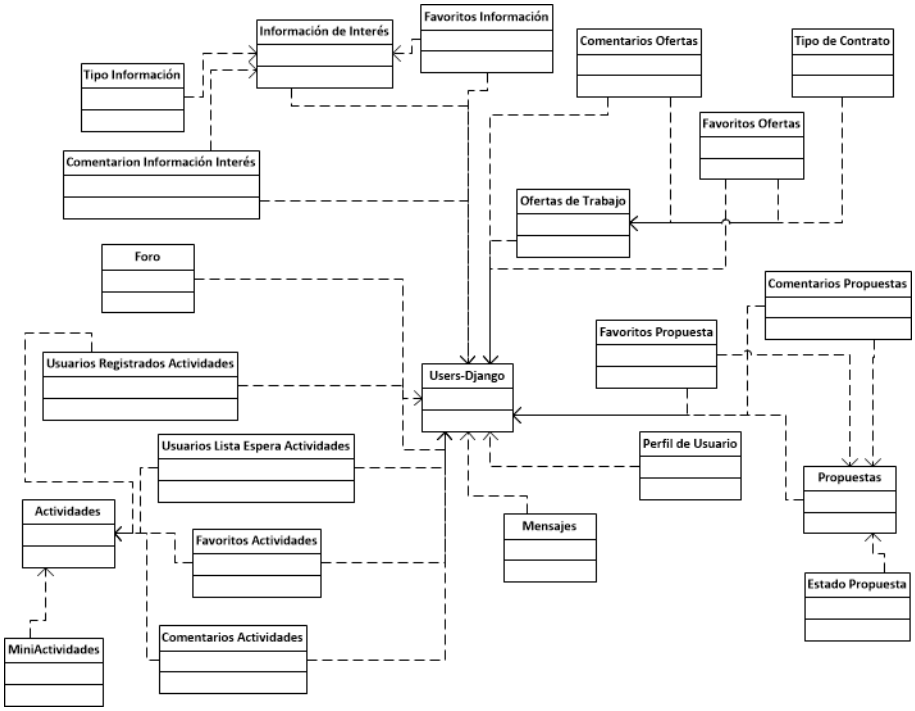

Figura 4.1: Base de datos

FIXME: la figura queda un poco rara, porque no se pueden ver los campos de cada tabla.

A continuación se enumeran y describen cada una de las aplicaciones que utilizan la base de datos mostrada en la figura 1:

- Theme: en esta aplicación se define la estructura que tendrá la *homepage*, también incluyen modelos para la creación de FAQs y la creación de un blog y sus respectivas entradas.
- Propuestas: con esta aplicación conseguimos que los usuarios propongan sus propias actividades o al tipo de actividades que desean asistir.
- Ofertas: con esta aplicación conseguimos dar a conocer las ofertas de trabajo y prácticas en empresas.
- Actividades: con esta aplicación conseguimos dar a conocer las distintas actividades que se realizan en la Universidad. Estas actividades van desde un simple seminario o charla hasta actividades un poco más grandes como podría ser un taller con varios seminarios dentro o incluso la organización y difusión de la semana de San Teleco.
- Información de interés: con esta aplicación conseguimos difundir el resto de información que se desee compartir como becas, cursos, concursos, etc.

- Perfil: con esta aplicación conseguimos complementar el modelo de usuario creado por Django. El modelo de Perfil creado debe referenciar a un usuario, pero no puede existir más de un perfil por usuario. No se mostrará públicamente algunos datos de carácter personal debido a la ley de protección de datos.

- BasicModels y BasicContent: estas dos aplicaciones han sido creadas para proporcionar una base para dos propósitos:

  - Crear páginas básicas como las creadas para mostrar las listas de ofertas, información de interés, etc.

  - o Extender el modelo al crear los modelos anteriormente comentados unificando en uno solo los campos comunes, además de proporcionar *templates* comunes para el resto de aplicaciones. Con estas dos aplicaciones conseguimos reducir el código, evitando en la medida de lo posible la repetición del mismo.

4.2.2. Tipo de usuarios

Dentro de la aplicación se distinguen tipos de usuarios, con distintos roles y diferentes permisos:

- No registrados: Estos usuarios sólo pueden ver el contenido de la aplicación pero no pueden interactuar ni con la aplicación ni con otros usuarios.

- Alumnos: Este tipo de usuarios además de ver el contenido, pueden interactuar con la aplicación, lo único que no pueden hacer es crear actividades, aunque podrían editarlas y eliminarlas si formaran parte de la administración en alguna de ellas.

- Profesores: Este tipo de usuarios además de lo que los alumnos pueden hacer también pueden crear actividades y cambiar el estado de una propuesta.

- Administradores: Además de todo lo descrito anteriormente. Tiene acceso a la parte de administración.

4.2.3. Amazon Web Service + Apache

La aplicación se ha desplegado en una máquina EC2 de Amazon Web Service usando una cuenta gratuita para estudiantes de duración de un año.

Para crear la máquina se ha escogido las siguientes opciones:

- El sistema operativo es: Ubuntu 14.04 LTS con una capacidad total de 8GB.

- Una instancia de tipo t2.micro: 1 GiB de memoria, 1 vCPU solo para EBS y plataforma de 32 bits o 64 bits. Las instancias T2 son instancias de desempeño con ráfagas que proporcionan un nivel base de desempeño de la CPU con la posibilidad de alcanzar ráfagas por encima del nivel básico. Las instancias de este tipo son ideales para aplicaciones que no usan la CPU por completo, a menudo o de manera constante, pero que de vez en cuando tienen que alcanzar ráfagas (por ejemplo, servidores web, entornos para desarrolladores y bases de datos). En este caso es una máquina de 64bits.

En las siguientes imágenes se pueden ver las diferentes características de la máquina creada:

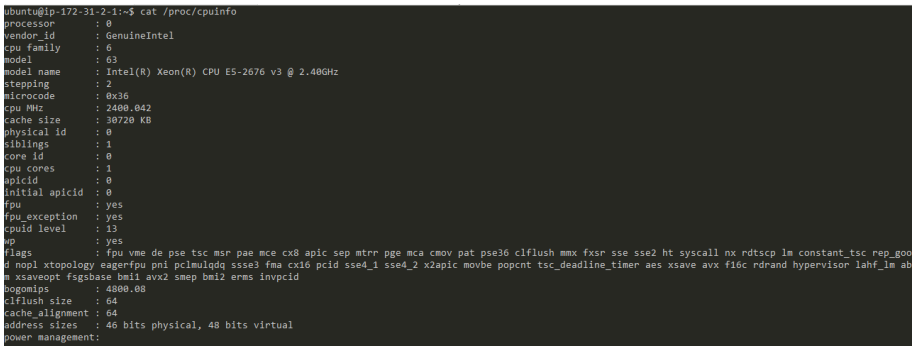

Figura 4.2: Salida de la ejecución del comando cat /proc/cpu/info

```
ubuntu@ip-172-31-2-1:~$ sudo dmidecode -t 4
# dmidecode 2.12
SMBIOS 2.4 present.

Handle 0x0401, DMI type 4, 26 bytes
Processor Information
        Socket Designation: CPU 1
        Type: Central Processor
        Family: Other
        Manufacturer: Intel
        ID: F2 06 03 00 FF FB 89 17
        Version: Not Specified
        Voltage: Unknown
        External Clock: Unknown
        Max Speed: 2400 MHz
        Current Speed: 2400 MHz
        Status: Populated, Enabled
        Upgrade: Other
```

Figura 4.3: Salida de la ejecución del comando sudo demidecode -t 4

```
ubuntu@ip-172-31-2-1:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 14.04.5 LTS
Release:        14.04
Codename:       trusty
```

Figura 4.4: Salida de la ejecución del comando lsb_release -a

```
ubuntu@ip-172-31-2-1:~$ lscpu
Architecture:            x86_64
CPU op-mode(s):          32-bit, 64-bit
Byte Order:              Little Endian
CPU(s):                  1
On-line CPU(s) list:     0
Thread(s) per core:      1
Core(s) per socket:      1
Socket(s):               1
NUMA node(s):            1
Vendor ID:               GenuineIntel
CPU family:              6
Model:                   63
Stepping:                2
CPU MHz:                 2400.042
BogoMIPS:                4800.08
Hypervisor vendor:       Xen
Virtualization type:     full
L1d cache:               32K
L1i cache:               32K
L2 cache:                256K
L3 cache:                30720K
NUMA node0 CPU(s):       0
```

Figura 4.5: Salida de la ejecución del comando lscpu

```
ubuntu@ip-172-31-2-1:~$ sudo lshw -class processor
  *-cpu:0
       description: CPU
       product: Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
       vendor: Intel Corp.
       physical id: 1
       bus info: cpu@0
       slot: CPU 1
       size: 2400MHz
       capacity: 2400MHz
       width: 64 bits
       capabilities: fpu fpu_exception wp vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp x86
-64 constant_tsc rep_good nopl xtopology eagerfpu pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdra
nd hypervisor lahf_lm abm xsaveopt fsgsbase bmi1 avx2 smep bmi2 erms invpcid
  *-cpu:1
       description: CPU
       vendor: Intel
       physical id: 4
       bus info: cpu@1
       slot: CPU 1
       size: 2400MHz
       capacity: 2400MHz
```

Figura 4.6: Salida de la ejecución del comando lshw

```
ubuntu@ip-172-31-2-1:~$ cat /proc/meminfo
MemTotal:        1016284 kB
MemFree:          262168 kB
Buffers:          111688 kB
Cached:           426432 kB
SwapCached:            0 kB
Active:           416344 kB
Inactive:         226900 kB
Active(anon):     105208 kB
Inactive(anon):      528 kB
Active(file):     311136 kB
Inactive(file):   226372 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:             0 kB
SwapFree:              0 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:        105124 kB
Mapped:            10540 kB
Shmem:               612 kB
Slab:              93192 kB
SReclaimable:      81436 kB
SUnreclaim:        11756 kB
KernelStack:        1488 kB
PageTables:         3100 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:      508140 kB
Committed_AS:     686388 kB
VmallocTotal:   34359738367 kB
VmallocUsed:        4408 kB
VmallocChunk:   34359726540 kB
HardwareCorrupted:     0 kB
AnonHugePages:     51200 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:       26624 kB
DirectMap2M:     1153024 kB
```

Figura 4.7: Salida de la ejecución del comando cat /proc/meminfo

Para la configuración de Apache, primero se debe instalar usando el comando:

```
apt-get install apache2
```

Para poder servir aplicaciones Django desde un servidor Apache, lo más habitual es añadir el módulo mod_wsgi, que permite servir aplicaciones hechas en Python, que tengan soporte para la interfaz WSGI (Como es el caso de Django y en este caso también de Mezzanine).

Para ello se instala el paquete del repositorio:

```
sudo apt-get install libapache2-mod-wsgi
```

A continuación se configuró el host virtual de Apache. Un virtual host permite mantener múltiples nombres de host en Apache. Gracias a ello, se puede decirle a Apache que redirija las peticiones procedentes de una URL determinada, a una aplicación concreta.

Para generar la configuración del host virtual, se creó un archivo en /etc/apache2/sites-available/ con el nombre de Welpe.

La configuración usada se muestra en la siguiente captura:

```
ubuntu@ip-172-31-2-1:~$ cat /etc/apache2/sites-available/Welpe.conf

<VirtualHost *:80>
        ServerName http://ec2-54-71-52-158.us-west-2.compute.amazonaws.com
        ServerAlias 54.71.52.158
        ServerAdmin welpe.tfg@gmail.com
        Alias /static /home/ubuntu/Welpe/static
        <Directory /home/ubuntu/Welpe/static>
                Require all granted
        </Directory>
        <Directory /home/ubuntu/Welpe/Welpe>
            <Files wsgi.py>
                Require all granted
            </Files>
        </Directory>

        WSGIDaemonProcess Welpe python-path=/home/ubuntu/Welpe:home/ubuntu/Welpe/Welpe/venv/lib/python2.7/site-packages
        WSGIProcessGroup Welpe
        WSGIScriptAlias / /home/ubuntu/Welpe/Welpe/wsgi.py
</VirtualHost>
```

Figura 4.8: Fichero de configuración en Apache

Después de configurar el anterior fichero, se activó el host virtual y se reinició el servidor mediante:

```
sudo a2ensite Welpe
sudo /etc/init.d/apache2 restart
```

4.3. Diseño e implementación del cliente

El cliente puede acceder a la interfaz web a través de un navegador web. Como bien se ha comentado anteriormente, se han utilizado distintas tecnologías para que la interfaz sea más

amigable. HTML5 y CSS3 junto con Bootstrap y jQuery han sido necesarias para definir cómo se visualizan y con qué estilos y diseños aparecen cada uno de los elementos visibles en esta aplicación.

Estaría bien un diagrama con las vistas que tiene.

Todas las vistas de este proyecto siguen el mismo esquema:

FIXME: estaría bien tener este esquema en un diagrama.

- Una barra superior o *header* que consta del logotipo de la aplicación junto con el login y un formulario de búsqueda. Además también consta del menú y ya por último se ha introducido también otra pequeña cabecera con el nombre de vista junto con lo que llamaríamos *breadcrumb* o migas de pan. Breadcrumb es un elemento que aporta al usuario la información de dónde se encuentra en cada momento dentro de un sitio web. Se indica la ubicación exacta de la página y la relación jerárquica de esta con respecto a la de inicio del sitio web.

- Contenido principal de la página. En esta sección lo que nos encontramos es con el contenido renderizado que estamos pidiendo a través de la URL introducida ya sea el perfil de usuario, una actividad, una oferta o cualquier otra petición.

- Footer. En el footer podemos encontrar el copyright®, política de cookies, etc.

4.3.1. Vistas de las páginas

A continuación se describen una a una todas las páginas de la aplicación explicando cada elemento que aparece en las páginas así como su utilidad. Todas las páginas en principio son públicas y accesibles para todo tipo de usuarios tanto para los registrados como para los no registrados aunque también se ha de decir, que estando registrado en esta aplicación, se puede interactuar por ejemplo escribiendo comentarios, creando propuestas, etc., además si en un futuro esta aplicación se expandiera podría tener contenido exclusivo sólo para usuarios registrados.

Página de inicio

La página de inicio es enviada al cliente cuando se recibe una petición GET sobre el recurso /, a través de la vista home. En esta vista hace un resumen de las características que tiene la aplicación web.

Figura 4.9: Hompage

Página del Perfil de Usuario

La página del perfil es enviada al cliente cuando se recibe una petición GET sobre el recurso /perfil/id_usuario.

En esta vista se muestra la información de un usuario.

El contenido dependerá de si un usuario accede a su propio perfil o al perfil de otro usuario.

Aquí nos encontramos varias cosas:

- El perfil de usuario que además de la información del usuario, se podrá interactuar con él, enviándole un mensaje privado.

- Si se accede al propio perfil, se podrá además de editarlo, tener acceso a la lista de mensajes tanto enviados como recibidos, a la lista de favoritos ordenados por cada uno de los eventos y ver las actividades en las que se desea participar.

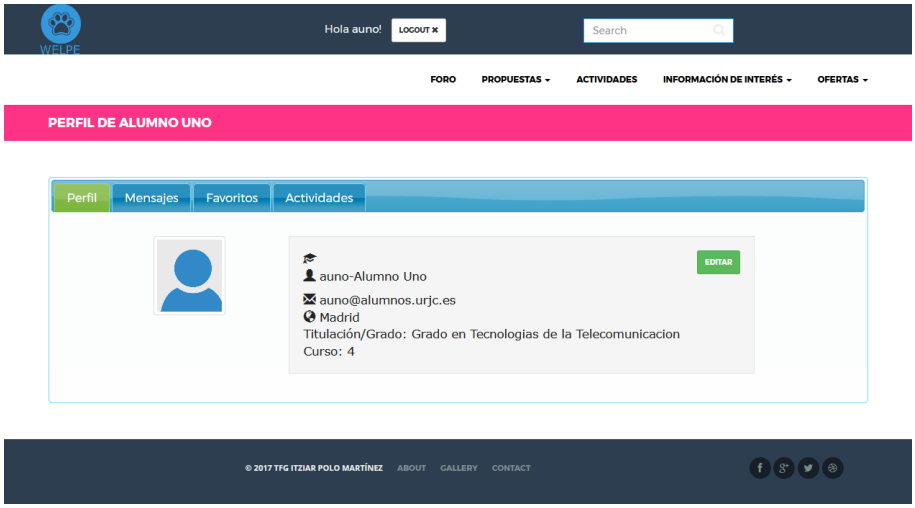

Figura 4.10: Vista del perfil de usuario

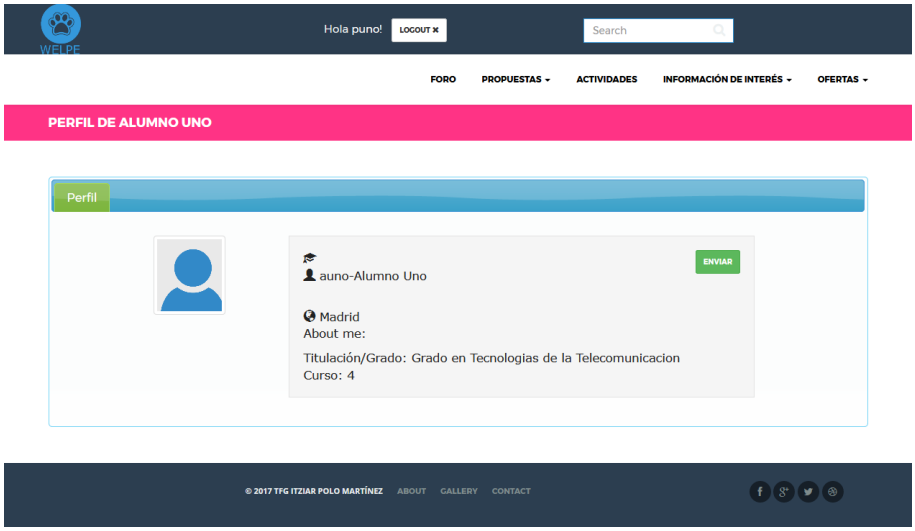

Figura 4.11: Vista del perfil de otro usuario distinto

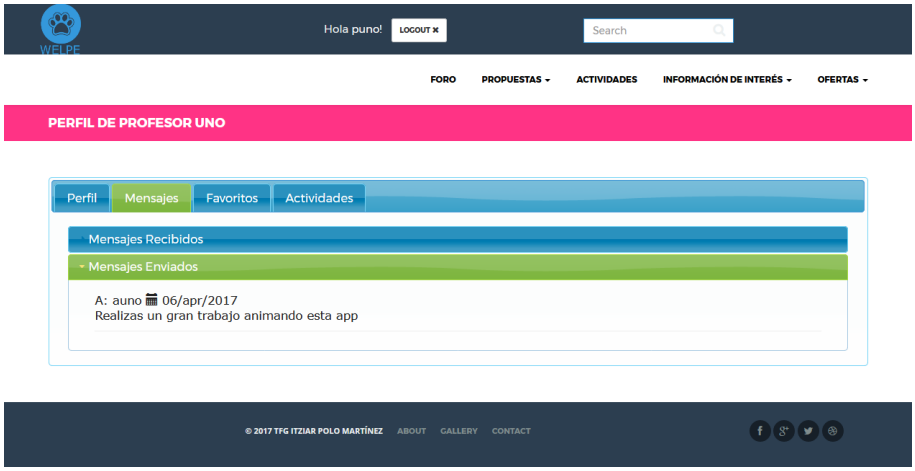

Figura 4.12: Vista de los mensajes

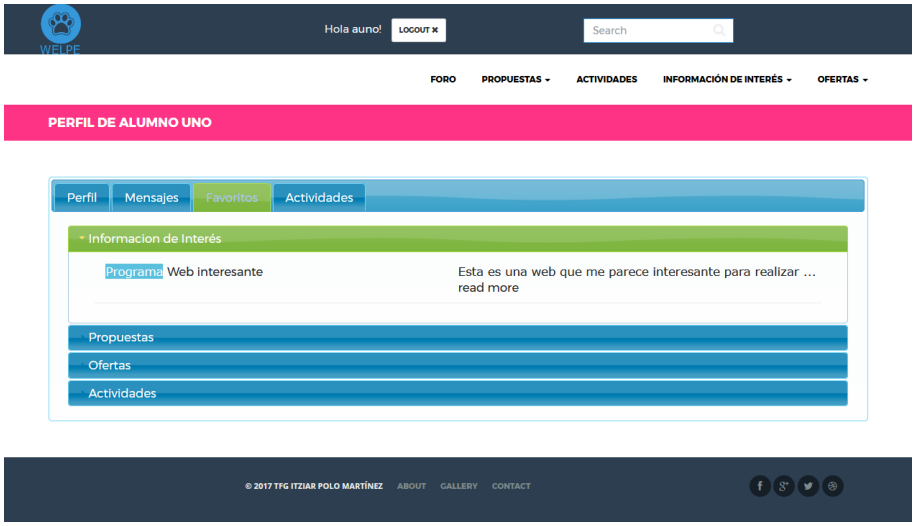

Figura 4.13: Vista de favoritos

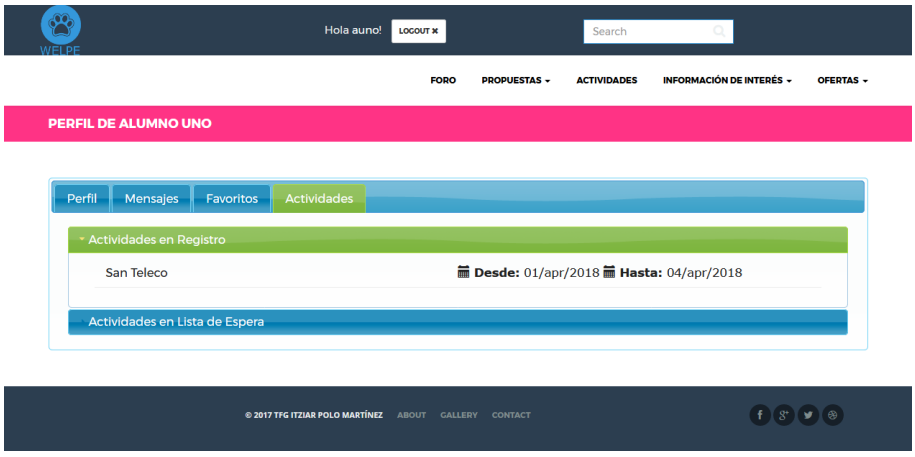

Figura 4.14: Vista de actividades registradas

Página del Foro

La página del foro es enviada al cliente cuando se recibe una petición GET sobre el recurso /foro.

La página foro contiene todos los mensajes publicados en el foro.

Cada mensaje contiene la foto y el nombre del usuario (aunque estos datos pueden ser anónimos), la fecha en la que se publicó, un título de comentario, una descripción (que será opcional).

Encima de estos mensajes al igual que en las demás vistas que contienen comentarios aparece un formulario para si se desea dejar un comentario.

El objetivo inicial del foro era dotar a la aplicación de un aspecto más social según lo comentado en los objetivos.

La función del foro es la de conectar a todos los usuarios, permitiendo un entorno universitario conectado y permitiendo mejorar el mismo, por ejemplo compartiendo curiosidades que los usuarios vayan encontrando durante su estancia en la universidad.

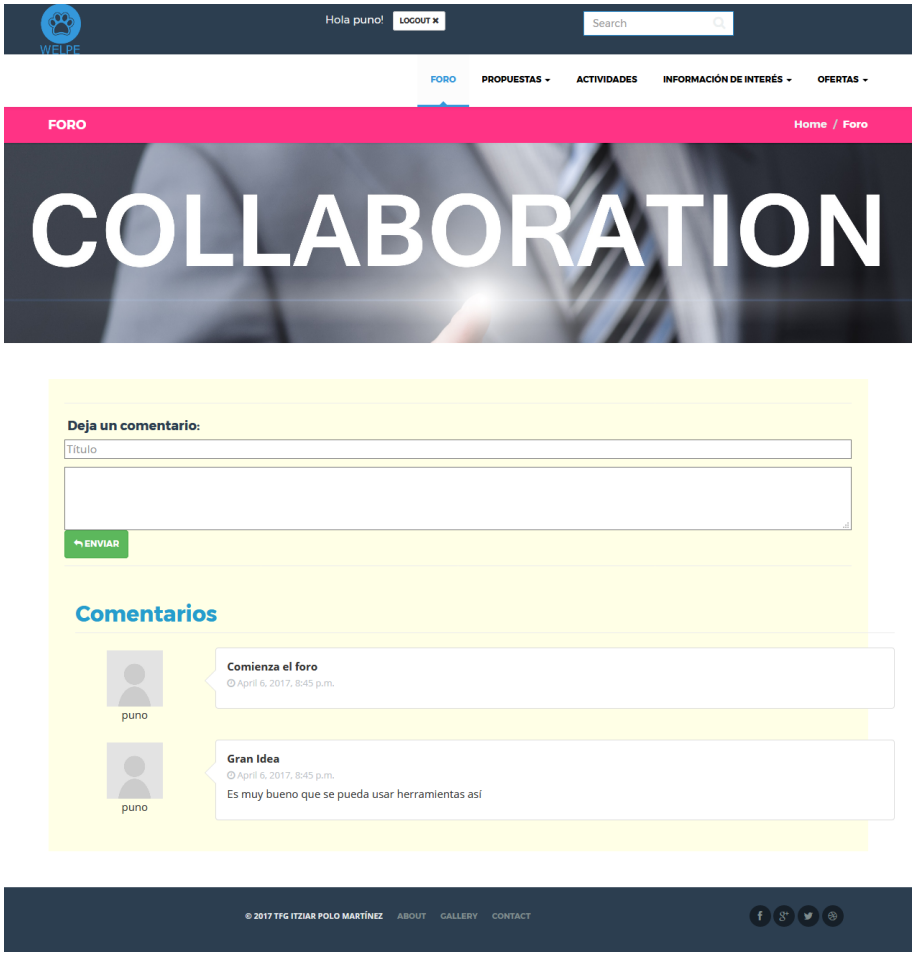

Figura 4.15: Vista del foro

Página de Propuestas

La página del listado de propuestas es enviada al cliente cuando se recibe una petición GET sobre el recurso /propuestas.

En esta vista se proporciona un listado con todas las propuestas que se han ido recopilando, junto con una pequeña parte de la descripción. Además a la izquierda del título se muestra el estado de la propuesta y a la derecha se puede observar varios elementos. En primer lugar un corazón que viene a significar si la propuesta se ha guardado en la lista de favoritos o no, además de un número que irá creciendo o decreciendo dependiendo del número de usuarios la guarden en sus listas de favoritos. A continuación se observa el número de comentarios que contiene cada una de las ofertas.

También cualquier usuario (registrado o no) puede filtrar las propuesta por las palabras que contenga el título o contenido de la propuesta, además también se puede filtrar por el estado de

las propuestas a elegir entre enviado, aceptado o rechazado, además si el usuario está registrado podrá filtrar si la propuesta está o no en su lista de favoritos.

Para introducir una nueva propuesta cualquier usuario registrado puede hacerlo. Para ello, se rellena el formulario con los datos correspondientes y envía ese formulario. Esta petición llega al servidor mediante una petición POST donde se encargará de sacar los datos y guardarlos correctamente en la base de datos, devolviendo como resultado una nueva vista donde se muestra la nueva propuesta.

Por último, sólo si el usuario ha creado una determinada propuesta o es un profesor o es el administrador: podrá modificar y/o eliminar dicha propuesta mediante los botones edit y delete.

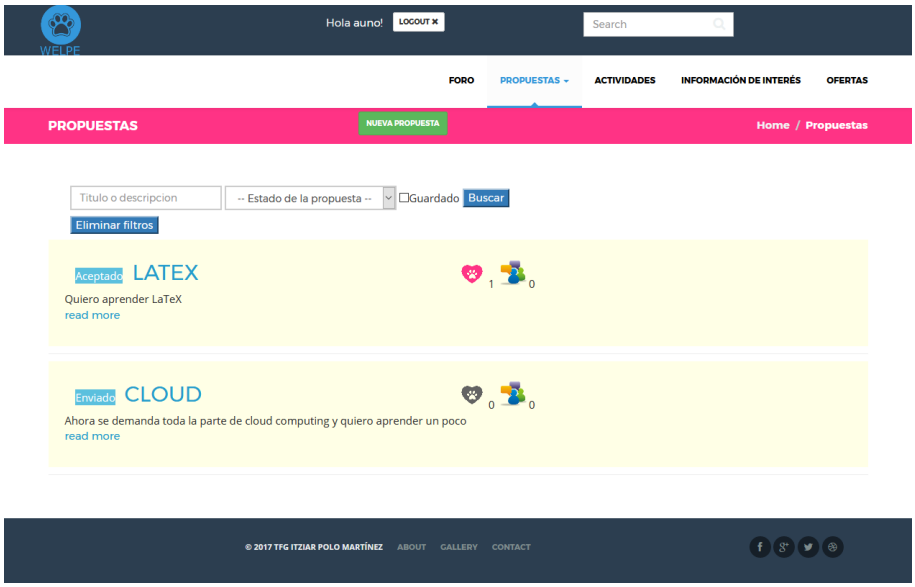

Figura 4.16: Vista de la lista de propuestas

Página de una Propuesta

La página de una propuesta es enviada al cliente cuando se recibe una petición GET sobre el recurso /propuestas/título_propuesta

En esta vista se muestra toda la información relativa a la propuesta previamente introducida a través del formulario comentado anteriormente.

Relativo a esta vista, se encuentra el nombre, el estado de la propuesta, número de comentarios, los comentarios, la popularidad que se mide en el número de favoritos, etc.

En esta vista se puede interactuar de las siguientes formas sólo si es un usuario registrado:

- Se puede crear una nueva propuesta mediante el botón nueva propuesta donde se abrirá un modal en el que se pedirán ciertos datos; este formulario ya cumplimentado se enviará al servidor donde se procesará y almacenara devolviendo como resultado una nueva vista donde se muestra la nueva propuesta que el usuario ha creado.

- Si el usuario es el propietario de la propuesta podrá modificarla mediante el botón edit y también podrá eliminarla mediante el botón eliminar.

- Junto con todo ello, está también el botón de favoritos donde el usuario puede agregar o eliminar la propuesta de su lista de favoritos.

- Se ha proporcionado un apartado de comentarios a la propuesta para que los usuarios puedan comentar cada propuesta.

- Sólo los profesores y el administrador pueden cambiar el estado de la propuesta. Esto es así, ya que el objetivo de las propuestas es que los alumnos den opinión de qué es lo que quieren aprender en los seminarios y/o proponer ellos mismos los seminarios que quieren impartir.

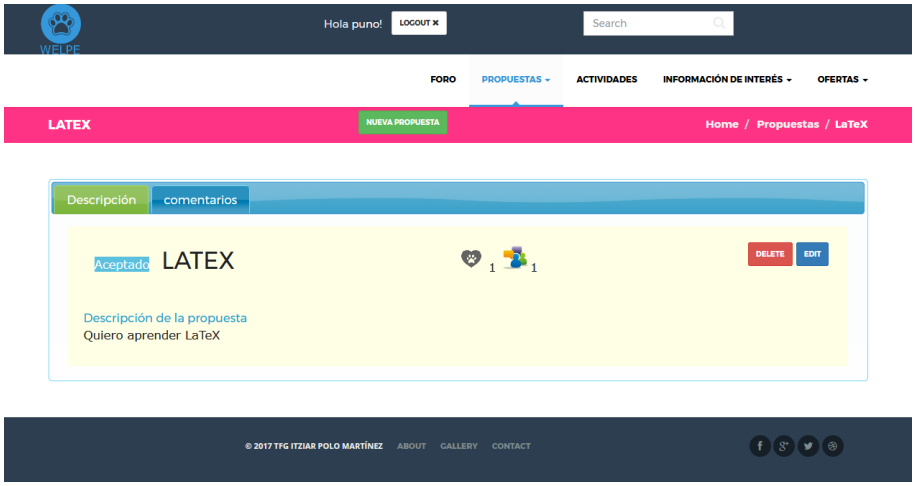

Figura 4.17: Vista de una propuesta

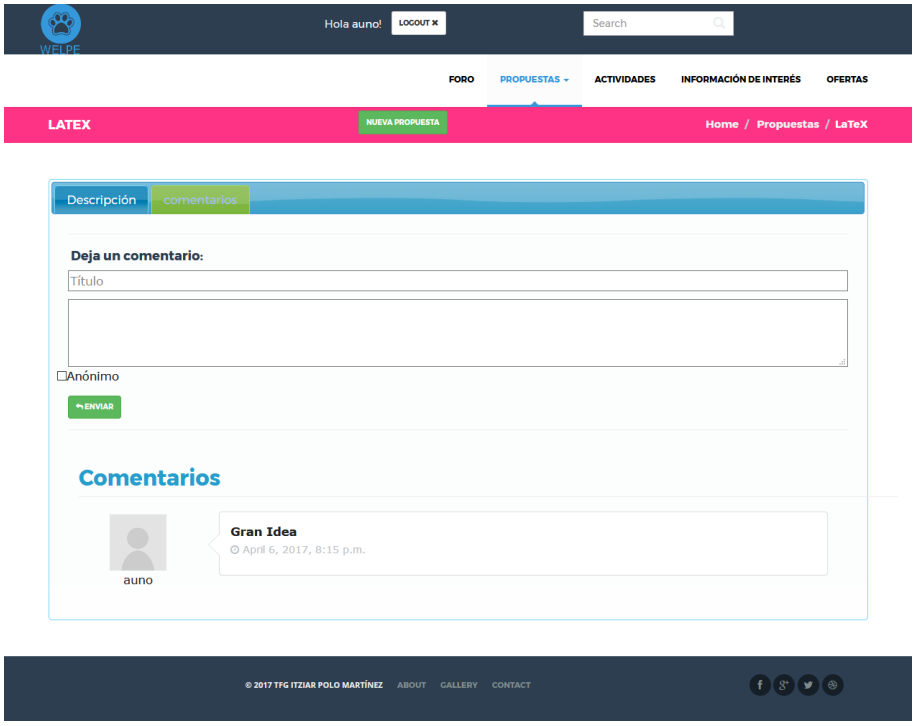

Figura 4.18: Vista de los comentarios a una propuesta

Página de Actividades

La página del listado de actividades es enviada al cliente cuando se recibe una petición GET sobre el recurso /actividades.

En esta vista se muestra un listado con todas las actividades que han ido recopilando junto con una pequeña parte de la descripción. Se puede observar el título junto con varios elementos a la derecha. En primer lugar un corazón que viene a significar si la actividad se ha guardado en la lista de favoritos o no, además de un número que irá creciendo o decreciendo dependiendo del número de usuarios que guarden en sus listas de favoritos dichas actividades. A continuación se observa el número de comentarios que contiene cada una de las actividades.

También cualquier usuario (registrado o no) puede filtrar las actividades por las palabras que contenga el título o contenido de la actividad, además si el usuario está registrado, podrá filtrar el listado de las actividades, si la actividad está o no en su lista de favoritos.

Para introducir una nueva actividad, sólo los administradores y los profesores pueden hacerlo. Esto es así, ya que se necesita el consentimiento de un profesor para que un alumno pueda organizar una actividad. Para ello se rellena el formulario con los datos correspondientes y se envía ese formulario. Esta petición llega al servidor mediante una petición POST donde se en-

cargará de sacar los datos y guardarlos correctamente en la base de datos, devolviendo como resultado una nueva vista donde se muestra la nueva actividad.

Por último, sólo si el usuario ha creado o está incluido en la lista de administradores de una determinada actividad o es un profesor o es el administrador: podrá modificar y/o eliminar dicha actividad mediante los botones edit y delete.

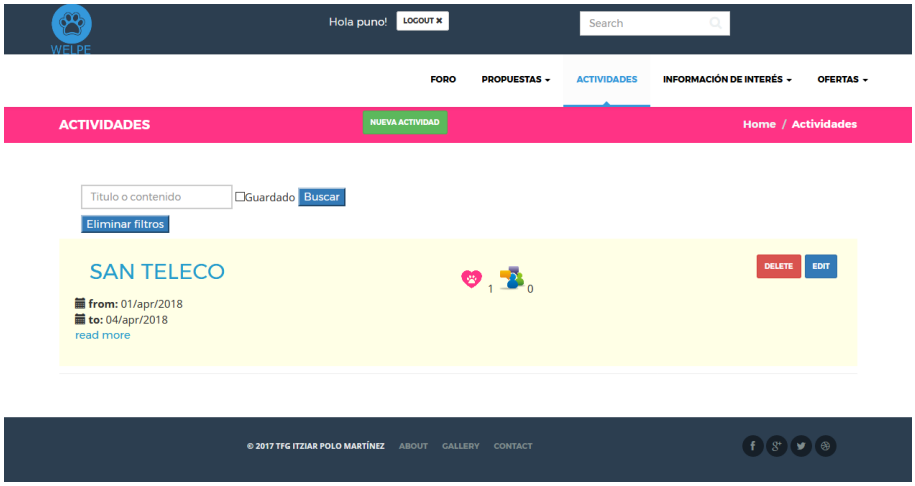

Figura 4.19: Vista de la lista de actividades

Página de una Actividad

La página de una actividad es enviada al cliente cuando se recibe una petición GET sobre el recurso /propuestas/título_actividad

En esta vista se muestra toda la información relativa a la actividad previamente introducida a través del formulario comentado anteriormente.

Relativo a esta vista, se encuentra el nombre, numero de comentarios, los comentarios, la popularidad de la actividad basada en el número de favoritos, la descripción de la actividad, las fechas, etc.

En esta vista se puede interactuar de las siguientes formas, sólo si es un usuario registrado:

- Si la actividad incluye que se puedan incluir miniactividades. Se podrá crear miniactividades mediante el botón nueva miniactividad donde se abrirá un modal en el que se pedirán ciertos datos; este formulario ya cumplimentado se enviará al servidor donde se procesará y almacenara. Las miniactividades son actividades dentro de una gran actividad.

- Si la actividad proporciona un registro, la vista contendrá una pestaña de registro de usuarios. Donde los usuarios que deseen participar puedan inscribirse. Mientras que los administradores de la actividad pueden además ver y obtener la lista de participantes mediante la descarga de un archivo xls.

- Se ha proporcionado un apartado de comentarios a la actividad para que los usuarios puedan comentar cada actividad.

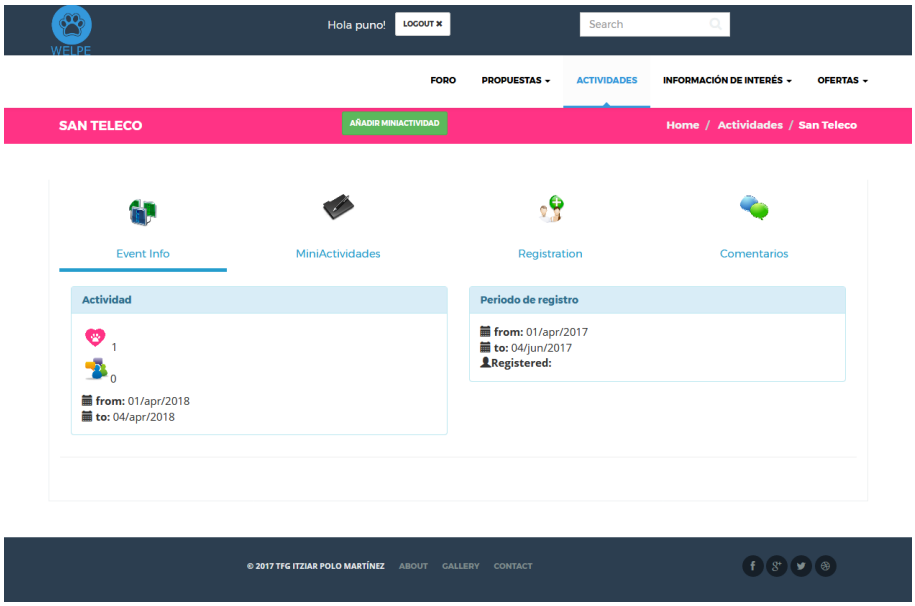

Figura 4.20: Vista de una actividad

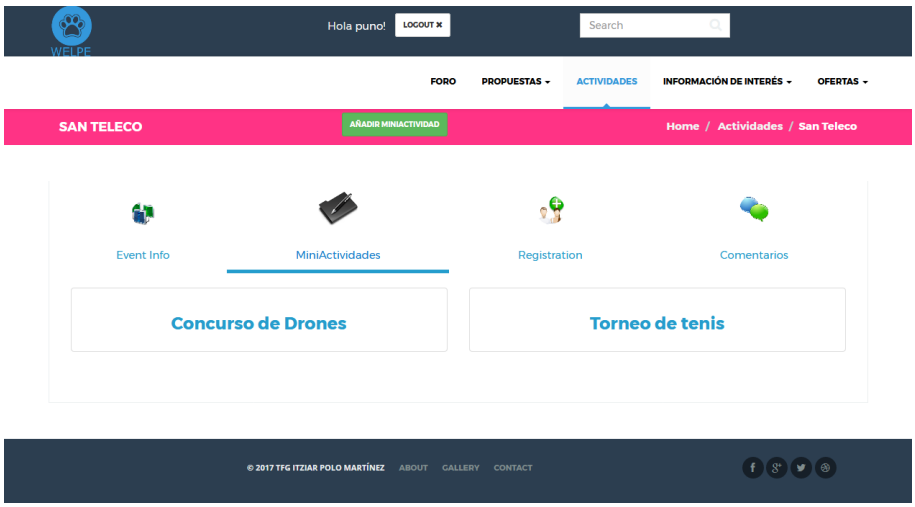

Figura 4.21: Vista de las miniactividades

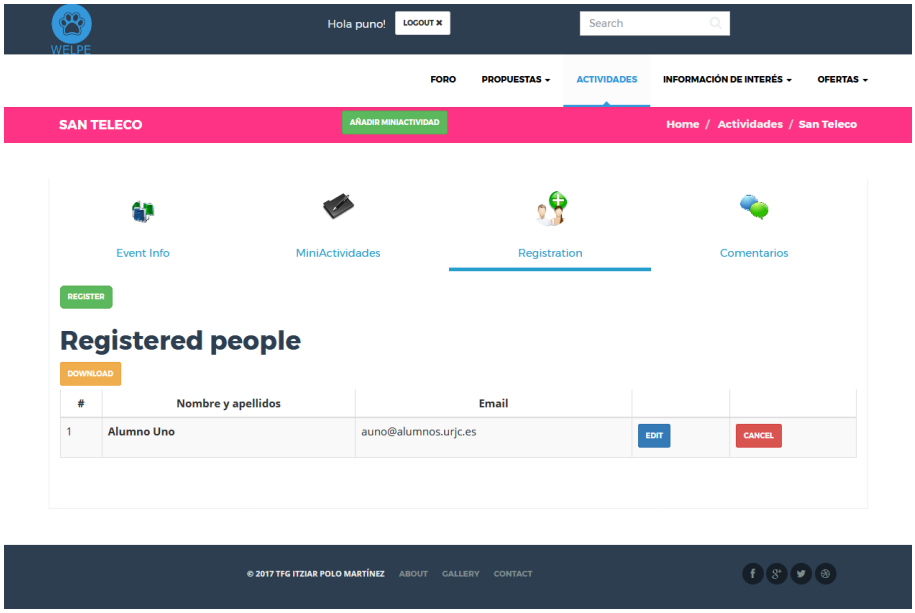

Figura 4.22: Vista del registro

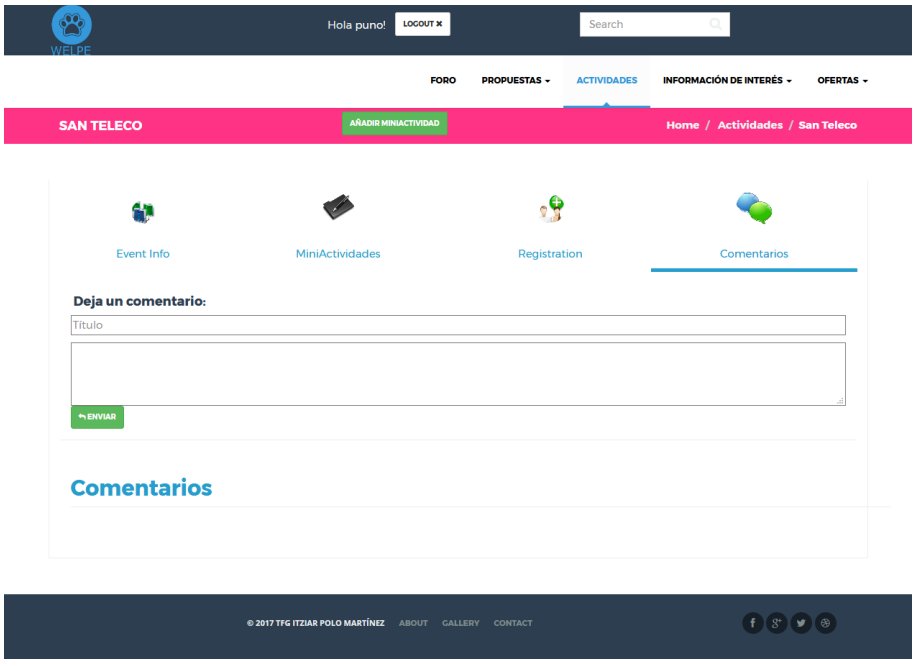

Figura 4.23: Vista de los comentarios a una actividad

Página de Ofertas

La página del listado de ofertas es enviada al cliente cuando se recibe una petición GET sobre el recurso /ofertas.

En esta vista se muestra un listado con todas las ofertas que se han ido recopilando junto con una pequeña parte de la descripción. Además a la izquierda del título se muestra el tipo de contrato que se ofrece y a la derecha se puede observar varios elementos. En primer lugar un corazón que viene a significar si la oferta se ha guardado en la lista de favoritos o no, además de un número que irá creciendo o decreciendo dependiendo del número de usuarios que guarden en sus listas de favoritos dichas ofertas. A continuación se observa el número de comentarios que contiene cada una de las ofertas.

También cualquier usuario (registrado o no) puede filtrar las ofertas por la/s palabra/s que contenga el título o contenido de la oferta, además también se puede filtrar por el tipo de contrato que se ofrece a elegir entre contrato indefinido, prácticas u otro tipo de contratos y además si el usuario está registrado podrá filtrar si la oferta está o no en su lista de favoritos.

Para introducir una nueva oferta cualquier usuario registrado puede hacerlo. Para ello se rellena el formulario con los datos correspondientes y envía ese formulario. Esta petición llega al servidor mediante una petición POST donde se encargará de sacar los datos y guardarlos correctamente en la base de datos, devolviendo como resultado una nueva vista donde se muestra la nueva oferta.

Por último, sólo si el usuario ha creado una determinada oferta o es un profesor o es el administrador: podrá modificar y/o eliminar dicha oferta mediante los botones edit y delete.

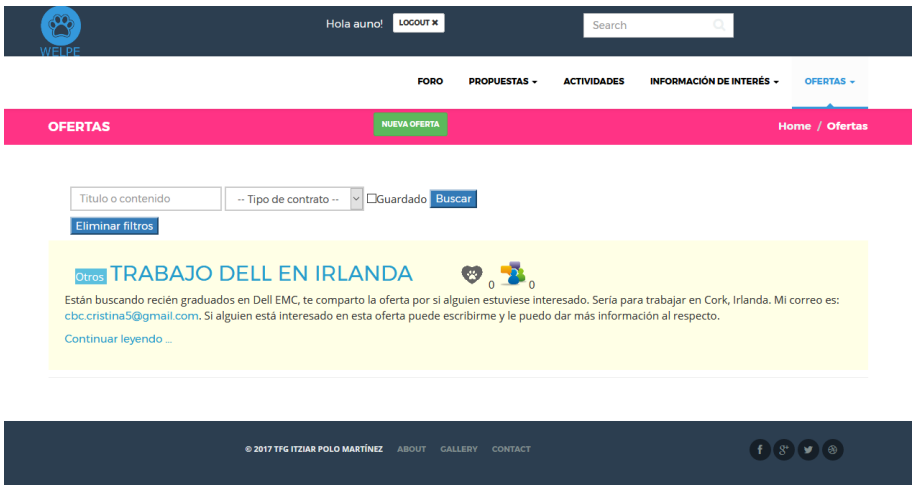

Figura 4.24: Vista de la lista de ofertas

Página de una Oferta

La página de una oferta es enviada al cliente cuando se recibe una petición GET sobre el recurso /ofertas/título_oferta

En esta vista se muestra toda la información relativa a la oferta previamente introducida a través del formulario comentado anteriormente.

Relativo a esta vista se encuentra el nombre, el tipo de contrato ofertado, número de comentarios, los comentarios, las veces que la oferta se ha guardado en la lista de favoritos, la descripción de la oferta, como aplicar a la oferta, etc.

En esta vista se puede interactuar de las siguientes formas sólo si el usuario registrado:

- Se puede crear una nueva oferta mediante el botón nueva oferta donde se abrirá un modal en el que se pedirán ciertos datos; este formulario ya cumplimentado se enviará al servidor donde se procesará y almacenara, devolviendo como resultado una nueva vista donde se muestra la oferta que el usuario previamente ha creado.

- Si el usuario es el propietario de la oferta podrá modificarla mediante el botón edit y también podrá eliminarla mediante el botón eliminar.

- • Junto con todo ello, está también el botón de favoritos donde el usuario puede agregar o eliminar la oferta de su lista de favoritos.

- Se ha proporcionado un apartado de comentarios a la oferta para que los usuarios puedan comentar cada oferta.

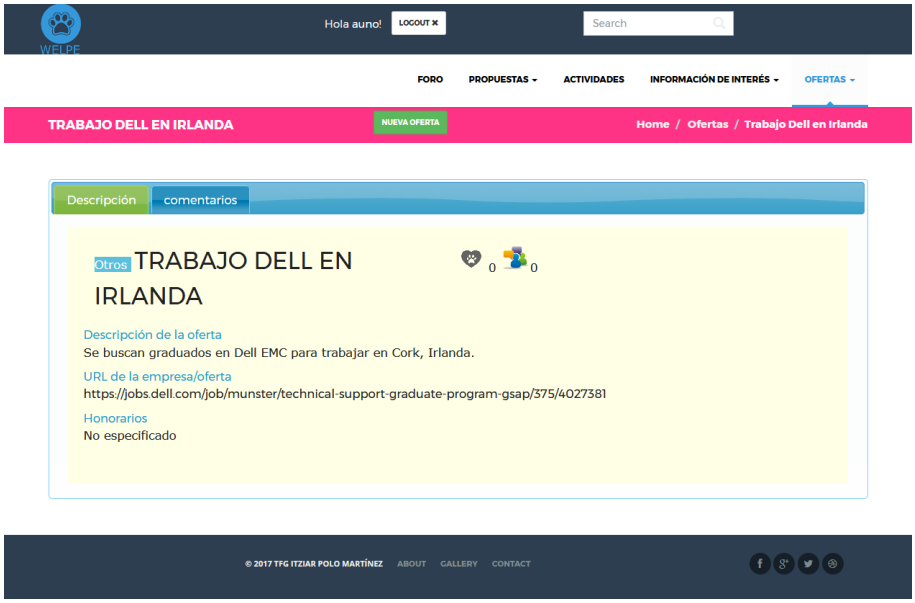

Figura 4.25: Vista de una oferta

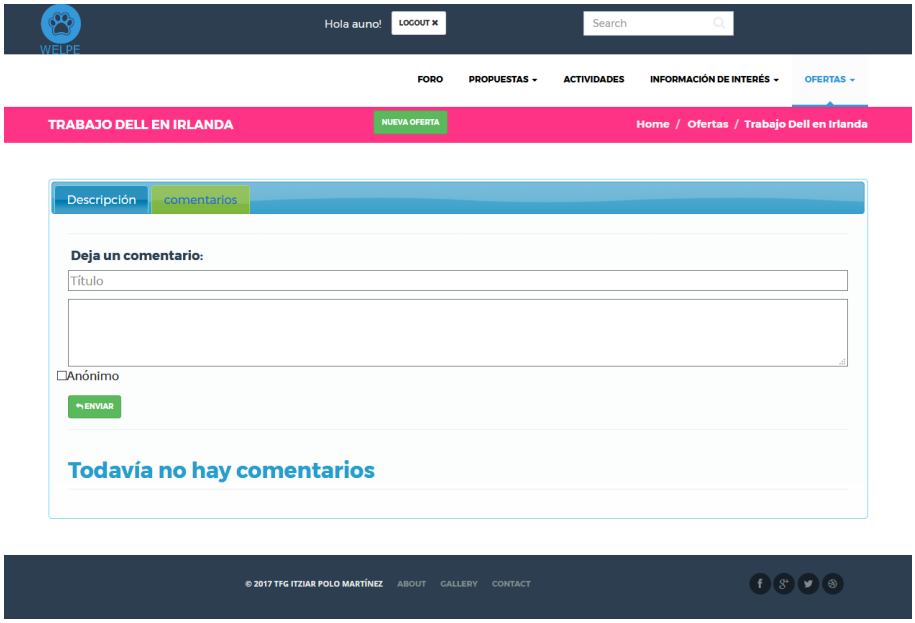

Figura 4.26: Vista de los comentarios a una oferta

Página de Información de Interés

La página del listado de informaciones es enviada al cliente cuando se recibe una petición GET sobre el recurso /informacion.

En esta vista se muestra un listado con toda la información de las distintas informaciones de interés que se han ido recopilando junto con una pequeña parte de la descripción. Además

a la izquierda del título se muestra la clase de información que ofrece y a la derecha se puede observar varios elementos. En primer lugar un corazón que viene a significar si la información de interés se ha guardado en la lista de favoritos o no, además de un número que irá creciendo o decreciendo dependiendo del número de usuarios que las guarden en sus listas de favoritos. A continuación se observa el número de comentarios que contiene cada información.

También cualquier usuario (registrado o no) puede filtrar este listado por la/s palabra/s que contenga el título o contenido de la información deseada, además también podemos filtrar por el tipo de información que se ofrece a elegir entre beca de estudios, programa, curso, concurso u otro tipo de información y además si el usuario está registrado podrá filtrar si la información está o no en su lista de favoritos.

Para introducir una nueva información cualquier usuario registrado puede hacerlo. Para ello se hace click en el botón "Añadir información" se rellena el formulario con los datos correspondientes y se envía. Esta petición llega al servidor mediante una petición POST donde se encargará de sacar los datos y guardarlos correctamente en la base de datos devolviendo como resultado una nueva vista donde se muestra la información de interés que el usuario previamente había enviado.

Por último, sólo si el usuario ha creado una determinada información de interés o es un profesor o es el administrador: podrá modificar y/o eliminar dicha información mediante los botones edit y delete.

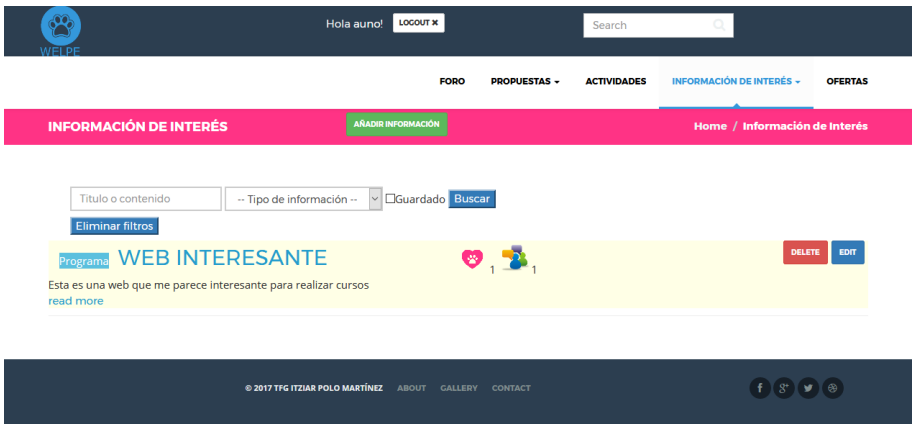

Figura 4.27: Vista de la lista de informaciones de interés

Página de una Información de interés

La página de una información es enviada al cliente cuando se recibe una petición GET sobre el recurso /informacion/título_información_de_interés

En esta vista se muestra toda la información relativa a una única información de interés que previamente ha sido incluida a través del formulario comentado anteriormente.

Relativo a esta vista se encuentra el título, el tipo de información ofrecida, número de comentarios, los comentarios, las veces que dicha información de interés se ha guardado en la lista de favoritos, la descripción de la información, url donde encontrar más información, etc.

En esta vista se puede interactuar de las siguientes formas sólo si es un usuario registrado:

- Se puede crear una nueva información mediante el botón añadir información donde se abrirá un modal en el que se pedirán ciertos datos, este formulario ya cumplimentado se enviará al servidor donde se procesará y almacenará devolviendo como resultado una nueva vista donde se muestra la nueva información de interés.

- Si el usuario es el propietario de la información de interés podrá modificar mediante el botón edit, y eliminarla mediante el botón delete.

- Junto con todo ello, está también el botón de favoritos donde el usuario puede agregar o eliminar dicha información de interés de su lista de favoritos.

- Se ha proporcionado un apartado de comentarios donde los usuarios pueden compartir sus opiniones y/o aclarar sus dudas acerca de la información de interés.

Figura 4.28: Vista de una información de interés

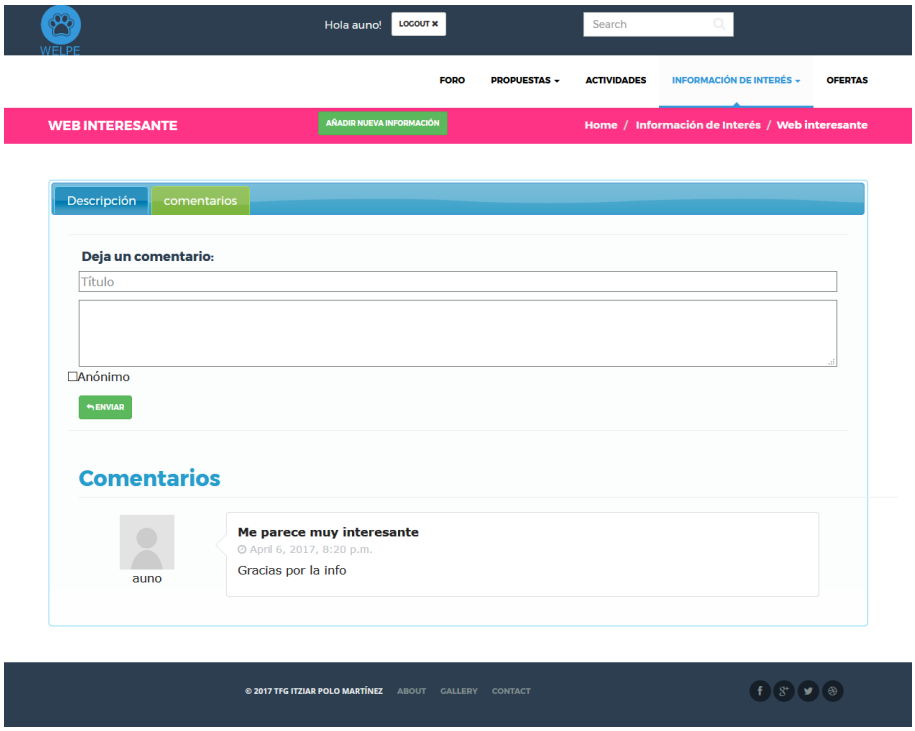

Figura 4.29: Vista de los comentarios a una información de interés

Página de Política de Cookies

Esta vista tiene que ver más con la legislación que con la propia aplicación.

El Real Decreto-ley 13/2012, de 30 de marzo, publicado en el «Boletín Oficial del Estado» el pasado sábado 31 de marzo de 2012, transpone la Directiva 2009/136/CE, del Parlamento Europeo y del Consejo, de 25 de noviembre de 2009, que se integra en la LSSI (Ley 34/2002, de 11

de julio, de servicios de la sociedad de la información y de comercio electrónico) modificando el punto segundo de su artículo 22, que queda redactado de la forma siguiente:

"Artículo 22.2 de la Ley 34/2002. Los prestadores de servicios podrán utilizar dispositivos de almacenamiento y recuperación de datos en equipos terminales de los destinatarios, a condición de que los mismos hayan dado su consentimiento después de que se les haya facilitado información clara y completa sobre su utilización, en particular, sobre los fines del tratamiento de los datos, con arreglo a lo dispuesto en la Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. Cuando sea técnicamente posible y eficaz, el consentimiento del destinatario para aceptar el tratamiento de los datos podrá facilitarse mediante el uso de los parámetros adecuados del navegador o de otras aplicaciones, siempre que aquél deba proceder a su configuración durante su instalación o actualización mediante una acción expresa a tal efecto. Lo anterior no impedirá el posible almacenamiento o acceso de índole técnica al solo fin de efectuar la transmisión de una comunicación por una red de comunicaciones electrónicas o, en la medida que resulte estrictamente necesario, para la prestación de un servicio de la sociedad de la información expresamente solicitado por el destinatario."

Por si no hubiera quedado claro del todo nos podríamos enfrentar a distintas sanciones dependiendo de la gravedad que consideren:

- Sanción Leve hasta 30.000 Euros. Art. 38.4 LSSI. Son infracciones leves: g) "El incumplimiento de las obligaciones de información o de establecimiento de un procedimiento de rechazo del tratamiento de datos, establecidas en el apartado 2 del artículo 22, cuando no constituya una infracción grave."

- Sanción Grave de 30.000 hasta 150.000 Euros. Art. 38.3 LSSI. Son infracciones graves: i) "El incumplimiento significativo de las obligaciones de información o de establecimiento de un procedimiento de rechazo del tratamiento de datos, establecidas en el apartado 2 del artículo 22."

Con lo cual lo más conveniente es informar al usuario para evitar posibles sanciones.

Páginas de Sitemaps

Para mejorar el posicionamiento SEO, una recomendación es tener dos sitemaps, uno para los buscadores (se puede acceder a través de /sitemap.xml) y otro creado especialmente para la

compresión humana (se puede acceder a través de /sitemap).

El primero de los sitemap (sitemap.xml) el cms que se utiliza (Mezzanine) lo crea por defecto.

El segundo (/sitemap), se ha creado especialmente una aplicación mostrar el recorrido del árbol.

Vista del panel de administración

Mezzanine al igual que Django, ofrece unas vistas para administrar la base de datos que es muy útil y fácil de usar. Desde ahí se puede crear, modificar o eliminar cualquier elemento que se desee y hacer un seguimiento de la base de datos.

Cada aplicación creada en este proyecto junto con otras, que el propio CMS posee, se encuentran disponibles en la parte de administración.

Podemos observar que además de las vistas explicadas anteriormente también podemos podemos crear otras que ya vienen predefinadas con Mezzanine como entradas para un Blog, un Page (que sería como una página simple).

Además tenemos otras configuraciones sobre la plataforma en sí, como las proporcionadas por los menus Site y Theme.

Esta vista sólo es accesible para los administradores de la plataforma. Se puede acceder a ella a través de /admin.

Capítulo 5

Evaluación del resultado

El presente proyecto pretendía ofrecer una nueva herramienta para la gestión y promoción de los distintos eventos fuera y dentro de la Universidad objetivo que se ha conseguido.

Analizando cada uno de los objetivos descritos en el capítulo 2 también obtenemos un resultado satisfactorio:

- Aplicación universal: se ha conseguido crear una aplicación compatible con los distintos dispositivos, sistemas operativos y navegadores web existentes. Además, también se adapta para ofrecer un control y apariencia adecuados independientemente del tipo de dispositivo utilizado.

- Sencillez en su uso: se ha creado un interfaz sencilla e intuitiva para todos los usuarios. Además presenta una página de contacto para poder comunicarse con el/los administrador/es.

- Gestionar y promocionar mejor las actividades para que los alumnos crezcan tanto a nivel personal como profesional. Además de las distintas actividades en cada una de las secciones, se ha proporcionado una herramienta para que los alumnos puedan proponer sus propias actividades (dentro de la sección propuesta) y los profesores si lo ven viable, esas propuestas se conviertan en futuras actividades. Con esta herramienta conseguimos que los profesores puedan escuchar lo que los alumnos más demandan.

- Web colaborativa: aunque de momento no hay muchos usuarios (los creados por mí), esta herramienta puede ser un complemento al actual Moodle o incluso en un futuro llegar a

sustituirlo añadiendo más módulos. Actualmente esta aplicación cuenta con la gestión de todos los eventos que se quieren hacer llegar a los alumnos y además consiguiendo algo que muy pocas veces ocurre y es que los alumnos puedan proponer sus ideas y salgan adelante en forma de seminarios, cursos, etc.

- Aportar un elemento social: se ha creado un foro para que todos los miembros de la universidad puedan comunicarse y colaborar para lograr un mejor entorno universitario.

- Uso de tecnologías web avanzadas: se han utilizado varias herramientas web como HTML5, CSS3, Python, Django, JS, etc. que juntándolas han propiciado un buen resultado final.

Capítulo 6

Conclusiones

6.1. Conocimientos aplicados

Para el desarrollo del presente proyecto he utilizado los conocimientos adquiridos durante mi etapa en la Universidad cursando el Grado en Ingeniería en Tecnologías de la Telecomunicación.

Durante mi etapa en la universidad he crecido como profesional pero también como persona. Puedo decir que he adquirido gran cantidad de conocimiento, con lo cual quedarme con una sola asignatura para reflejar este proyecto sería un poco injusto; ya que de cada asignatura me llevo aunque sólo sea una pequeña parte y esa parte al final queda reflejada en lo que se hace día a día.

Si tuviera que destacar alguna o algunas asignaturas serían las orientadas en el área de programación puesto que su contenido y conocimiento se ajustan más al contexto del presente trabajo:

- Fundamentos de la Programación (FDP): es la asignatura que proporciona la base de la programación. Es donde ves por primera vez cómo se usa "eso de los ordenadores" y donde te dan ganas de salir corriendo (pero no lo haces).

- Servicios y Aplicaciones Telemáticas (SAT): es la asignatura en que por primera vez ves que es útil eso de la programación y se pueden hacer cosas "chulas" e interesantes ya que en las anteriores asignaturas del área de programación eran mediante una terminal. En esta asignaturas aprendí Python, Django y también a crear webs. Con esta asignatura

también descubrí que realmente sí que me gustaba programar (antes de esta asignatura, para mí, las asignaturas de esta área eran una continua tortura)

- Ingeniería de Sistemas de Información (ISI): es la asignatura donde aprendes a desarrollar junto a tu equipo con herramientas que en el mundo profesional real se usan. A ver que hay vida más allá de las BBDD relacionales, etc. Aunque estas herramientas no las he usado en mi proyecto, las tengo muy presente en mi carrera profesional.
- Desarrollo de Aplicaciones Telemáticas (DAT): es una asignatura donde aprendes a comunicarte (y sobre todo pegarte) con distintas APIs.
- Del resto de asignaturas me quedo con la capacidad de esfuerzo y entrega que he tenido, intentando dar al máximo en cada una de ellas, algunas veces viendo recompensa otras veces había que seguir intentándolo con mucho más empeño y esfuerzo.

También he aprendido bastante de los seminarios impartidos por profesores e incluso me ha tocado a mí dar alguno. En el que impartí sobre LaTeX, aprendí un poco más sobre esa herramienta que para muchos es desconocida además aprendí el trabajo y esfuerzo que hay al organizar un seminario que van más allá de una simple presentación.

Aunque no se promocione mucho, también he aprendido bastante en los diferentes cursos llamados MOOCs (Massive Online Open Courses) que ofrecen las diferentes universidades del mundo y plataformas. Empezó siendo un entretenimiento y al final se convirtió en un apoyo para mis estudios (aunque hubiera cursado alguno que no fuera directamente de la rama 'Computer Science'). Con estos cursos además del propio contenido y de inglés, he aprendido a abrir mi mente y no estar encerrada en mi burbuja.

Quiero terminar este apartado con una GRACIAS a todos y cada uno de los profesores. De cada uno de vosotros me llevo una parte, tanto personal como profesionalmente. Seguro que aplicaré todos estos conocimientos proporcionados a lo largo de mi carrera profesional.

6.2. Conocimientos adquiridos

Durante la realización de este proyecto, no sólo he aprendido a poner en práctica mis conocimientos adquiridos durante la carrera, sino que este trabajo me ha permitido adquirir nuevos conocimientos y experiencias.

Al iniciar este trabajo, no me había enfrentado anteriormente a la creación de una aplicación de tal envergadura. Sin embargo, me he demostrado que mis conocimientos adquiridos durante la carrera (esfuerzo y constancia) estaban ahí que por mucho que diga "no puedo más" o "no lo voy a conseguir" siempre voy un poco más allá y siempre doy ese último pequeño esfuerzo.

En cuanto a mis conocimientos, he solidificado y madurado mis conocimientos en todas las tecnologías utilizadas y descritas en el capítulo 3.

También he aprendido sobre el mundo de los gestores de contenido. Lo útiles que pueden llegar a ser y lo rápido que crecen hoy en día.

Para el desarrollo he utilizado el IDE PyCharm y he decir que es una gran ayuda a la hora de depurar código y sobre todo a la hora de formatearlo (seguro que he infrautilizado este IDE pero en un futuro pienso utilizarlo mucho más).

Al decidirse desplegarlo en la nube, he adquirido conocimientos en *cloud* que seguro serán muy útiles en mi carrera profesional, además también me incita a aprender un más sobre este tema tan de moda y tan olvidado en la universidad.

6.3. Trabajos futuros

- Autenticación mediante LDAP de la Universidad
- Ampliación de esta aplicación con nuevas sub-aplicaciones para poder ser más completo y que en un futuro pueda sustituir a Moodle o al menos que puede equipararse al nivel de Moodle.
- Mejor eficiencia: siempre puede mejorarse la eficiencia de la aplicación, y debería mejorarse sobre todo las peticiones a la base de datos para así tener una mayor velocidad de navegación.
- Aplicación móvil: una aplicación móvil que ofrezca todas las funcionalidades que ofrece la aplicación web. Los dispositivos móviles son muy usados por cualquier persona joven y podría llegar a más alumnos. Para los más tradicionales, en cuanto a tecnologías se refiere, seguiría teniendo la aplicación web.

Bibliografía

[1] Página oficial de jQuery.

http://jquery.com/

[2] Página sobre JSON.

http://www.json.org/

[3] Página oficial de Bootstrap.

http://getbootstrap.com/

[4] Página oficial de Python.

https://www.python.org/

[5] Página oficial de Django.

https://docs.djangoproject.com/

[6] Página con las normas de estilo de Python

http://legacy.python.org/dev/peps/pep-0008/

[7] Página oficial de Mezzanine.

http://mezzanine.jupo.org/

[8] Página oficial de PyCharm

https://www.jetbrains.com/pycharm/

[9] Página sobre HTML5.

http://www.w3schools.com/html/html5_intro.asp

[10] Página sobre CSS3.

http://www.w3schools.com/css/css3_intro.asp/

[11] Información sobre la legislación española respecto a la política de cookies http://politicadecookies.com/sanciones.php

[12] Amazon Web Service versión estudiantes https://aws.amazon.com/es/education/awseducate/

[13] Despliegue en Apache en español http://blog.enriqueoriol.com/2014/06/lanzando-django-en-produccion-con.html

[14] Despliegue en Apache en inglés https://coderwall.com/p/ooerda/python-django-apache-ubuntu

[15] Obtener datos de la CPU en Linux http://www.binarytides.com/linux-cpu-information/

Apéndice A

Instalación y Uso

Para el funcionamiento de la aplicación lo primero que hay que hacer es instalar todos los paquetes necesarios. Estos paquetes se encuentran en el archivo "requirements.txt". Por lo tanto con ejecutar en una *shell* "pip install –r requirements.txt" se instalaran los paquetes necesarios, sino es que ya están instalados en la máquina.

Una vez instalados, ejecutar en una *shell* el comando "python manage.py createdb", que creará una base de datos, creando además un superusuario-administrador. A continuación hay que ejecutar el comando "python manage.py runserver". Cuando la aplicación esté en funcionamiento nos dirá tanto la IP como el PUERTO, por defecto localhost 8000. Se abrirá un navegador web donde habrá que introducir la URL "htttp://localhost:8000"

Apéndice B

Publicación

La aplicación se encuentra publicada en: `http://ec2-54-71-52-158.us-west-2.compute.amazonaws.com/` La cual estará disponible durante el año 2017.

El código fuente se encuentra disponible en: `https://github.com/itziar/Welpe/`

La presente memoria se encuentra disponible en: `https://github.com/itziar/TFG-Memoria/`