



GRADO EN INGENIERÍA EN TECNOLOGÍA DE LA
TELECOMUNICACIÓN

Curso Académico 2015/2016

Trabajo Fin de Grado

SHARE ERASMUS: DESARROLLO DE UNA
APLICACIÓN WEB COLABORATIVA
ORIENTADA A ESTUDIANTES ERASMUS

Autor : Marcos Martín Vedriel

Tutor : Dr. Gregorio Robles Martínez

Trabajo Fin de Grado

SHARE ERASMUS: DESARROLLO DE UNA APLICACIÓN WEB
COLABORATIVA ORIENTADA A ESTUDIANTES ERASMUS

Autor : Marcos Martín Vedriel

Tutor : Dr. Gregorio Robles Martínez

La defensa del presente Trabajo Fin de Grado se realizó el día de
de 2016, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2016

*Dedicado a
mis padres*

Resumen

El desarrollo de este trabajo consiste en la creación de una aplicación web colaborativa orientada a estudiantes Erasmus de la ETSIT. El objetivo de la aplicación es proporcionar los mecanismos necesarios para que antiguos estudiantes aporten información útil a los futuros estudiantes Erasmus, para así crear una comunidad donde se comparta información y experiencias de este programa europeo.

La aplicación web ha sido desarrollada principalmente con el uso de tecnologías como *HTML5*, *CSS3*, *Javascript* y su *framework Angular JS* en el lado del cliente, y con el *framework Django* de *Python* en el lado del servidor.

Debido a la importancia de seleccionar bien un destino europeo donde vivir las experiencias que proporciona el programa Erasmus, surge la necesidad de crear una herramienta como ésta que pueda ponerse en producción en la escuela.

Summary

The development of this work consists the creation of a collaborative web application aimed at Erasmus students ETSIT. The objective of the application is to provide the necessary mechanisms for former students provide useful information to future Erasmus students, in order to create a community where it shares information and experiences of this European program .

The web application has been developed mainly with the use of technologies such as *HTML5*, *CSS3*, *Javascript* and *Angular JS framework* on the client side, and the *framework Django Python* on the server side.

Due to the importance of selecting a good choice a European destination to live the experiences provided by the Erasmus program, it arises the need to create a tool like this that can be brought into production at school.

Acrónimos

API: Application Programming Interface

BSD: Berkeley Software Distribution

CSS: Cascading Style Sheets

ECTS: European Credit Transfer and Accumulation System

ERASMUS: EuRopean Community Action Scheme for the Mobility of University Students

ETSIT: Escuela Técnica Superior de Ingenieros de Telecomunicación

HTML: HyperText Markup Language

HTTP: Hypertext Transfer Protocol

MIT: Massachusetts Institute of Technology

SSH: Secure SHell

VPS: Virtual Private Server

W3C: World Wide Web Consortium

XML: eXtensible Markup Language

Índice general

Resumen	III
Summary	V
Acrónimos	VII
Índice de figuras	XI
1. Introducción	1
1.1. Motivación	1
1.2. ¿Qué es Erasmus?	1
1.3. Aplicaciones web similares	2
1.4. Estructura de la memoria	3
2. Objetivos	5
2.1. Objetivo general	5
2.2. Objetivos específicos	5
3. Estado del arte	7
3.1. Python	7
3.1.1. Django	7
3.2. Tecnologías web	8
3.2.1. HTML5	8
3.2.2. CSS3	9
3.2.3. Bootstrap	9
3.2.4. JavaScript	9

3.2.4.1. Angular JS	10
3.3. Alojamiento web	10
3.3.1. Github	10
3.3.2. OVH	11
4. Diseño e implementación	13
4.1. Arquitectura general	13
4.2. Diseño e implementación del servidor	14
4.2.1. Modelos de la base de datos	14
4.2.2. API	17
4.2.3. Backups	18
4.3. Diseño e implementación del cliente	19
4.3.1. Vistas de las páginas	20
4.3.2. Vistas de la configuración	28
4.3.3. Vista del panel de administración	33
5. Resultados	35
5.1. Pruebas en servidor local	36
5.2. Pruebas en servidor remoto	36
5.3. Resultado final	37
6. Conclusiones	39
6.1. Consecución de objetivos	39
6.2. Aplicación de lo aprendido	40
6.3. Lecciones aprendidas	41
6.4. Trabajos futuros	42
Bibliografía	43
A. Instalación y uso	45
B. Recurso para descargar la aplicación	47

Índice de figuras

4.1. Esquema de ficheros del proyecto	14
4.2. Archivo /etc/crontab	19
4.3. Página de inicio	21
4.4. Página de inicio de sesión y registro	22
4.5. Página de universidades	23
4.6. Página de una universidad	24
4.7. Página de una asignatura	26
4.8. Página de un usuario	27
4.9. Página de contacto	28
4.10. Página de política de privacidad	28
4.11. Página de configuración de la cuenta	29
4.12. Página de cambio de contraseña	30
4.13. Página de configuración de mis universidades	31
4.14. Página de configuración de mis asignaturas	32
4.15. Página de configuración de mis ciudades	33
4.16. Página de administración	34

Capítulo 1

Introducción

1.1. Motivación

La aplicación web desarrollada en este trabajo surge de la necesidad de crear una herramienta colaborativa en la que antiguos estudiantes puedan compartir información y experiencias con futuros estudiantes Erasmus.

Los estudiantes que van a estudiar a otra universidad con el programa Erasmus quieren estar informados de las asignaturas que se imparten en cada universidad, si son o no fáciles, etc. También les interesa conocer las ciudades, dónde alojarse, qué medios de transporte hay, qué nivel de vida, por dónde salen los universitarios, etc.

La ETSIT no proporciona plataformas específicas en las que los estudiantes Erasmus se intercambien información y experiencias, por lo que esta aplicación puede ser una solución que puedan disfrutar sus estudiantes.

1.2. ¿Qué es Erasmus?

Erasmus [1] es un programa de becas que promueve el intercambio de estudiantes entre universidades europeas. Los países que participan en este programa Erasmus son los estados miembros de la Unión Europea, además de Islandia, Liechtenstein, Noruega, Suiza, Turquía y

Macedonia. Actualmente está en vigor el programa llamado Erasmus+¹, que tiene vigencia para el periodo 2014-2020.

Las becas Erasmus+ (Plan de Acción de la Comunidad Europea para la Movilidad de Estudiantes Universitarios) pueden tener una duración de entre tres meses a un curso completo como máximo, aunque los alumnos pueden optar por esta beca durante 3 cursos. Para optar a una beca Erasmus es necesario estar cursando un título oficial de Grado/ Licenciatura/ Diplomatura/ Ingeniería o posgrado, haber superado el número de créditos mínimo exigidos por cada universidad, y pertenecer a cualquier estado que participe en el programa Erasmus o estar en posesión del permiso de residencia legal.

El programa Erasmus trata de impulsar las perspectivas laborales y el desarrollo personal, además de ayudar a los sistemas de educación, formación y juventud de cada uno de los países a proporcionar una enseñanza y un aprendizaje que doten a las personas de las capacidades necesarias para el mercado laboral y la sociedad actual y futura.

Pero Erasmus no es solo un programa puramente académico, sino que es una experiencia vital en la que los estudiantes pueden hacer amistades con gente de todas partes del mundo, conocer nuevas culturas y al fin y al cabo, a vivir de forma independiente nuevas experiencias fuera del hogar.

1.3. Aplicaciones web similares

Hay una aplicación web que en términos generales tiene el mismo objetivo que la aplicación que se desarrolla en este trabajo. Se trata de *erasmusu*². Tiene un formato más complejo, contiene la información más dividida en distintas páginas. Lo que se quiere conseguir con este trabajo es crear una aplicación web más sencilla que permita ofrecer, en términos generales, la misma información pero con un formato más comprimido y preciso.

¹<http://www.erasmusplus.gob.es/>

²<http://erasmusu.com/es>

Por otro lado existe una aplicación web sobre Erasmus desarrollada en un Proyecto Fin de Carrera (*TuErasmus: Diseño y desarrollo de una aplicación web colaborativa que sirva de apoyo para futuros estudiantes Erasmus*³) creada por Rawan Nazmi-Issa Khozouz, una alumna de la ETSIT. El presente TFG no es una continuación del Proyecto Fin de Carrera mencionado anteriormente, sino que se trata de otra aplicación distinta con una visión algo diferente, y proporcionando una interfaz más atractiva para los usuarios.

1.4. Estructura de la memoria

- **Capítulo 1. Introducción:** este capítulo expone el contexto del presente trabajo.
- **Capítulo 2. Objetivos:** este capítulo describe los objetivos para el desarrollo del presente trabajo.
- **Capítulo 3. Estado del arte:** este capítulo describe las distintas tecnologías utilizadas en la realización de la aplicación web.
- **Capítulo 4. Diseño e implementación:** en este capítulo se describe la aplicación diseñada. Se describe su estructura y su funcionamiento.
- **Capítulo 5. Resultados:** en este capítulo se analizan los resultados obtenidos.
- **Capítulo 6. Conclusiones:** en este capítulo se comentan las conclusiones finales tras la realización del trabajo y se comentan posibles líneas de desarrollo futuras.

³<https://gsyc.urjc.es/~grex/pfcs/2014-rawan-khozouz/>

Capítulo 2

Objetivos

2.1. Objetivo general

El objetivo principal de este trabajo es la creación de una aplicación web colaborativa orientada a alumnos que han estudiado o van a estudiar con una beca Erasmus. Dicha aplicación debe ofrecer información relevante sobre las asignaturas, universidades o ciudades de los diferentes destinos europeos.

2.2. Objetivos específicos

- **Estudiar las necesidades de los estudiantes:** para que una aplicación web tenga éxito, debe cubrir las necesidades de los visitantes objetivos de la web. Para ello es necesario conseguir un asesoramiento de antiguos estudiantes Erasmus para orientar el contenido de la web a las necesidades de los futuros Erasmus.
- **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*, *Angular JS* y *Django* y aprovechar sus funcionalidades.
- **Web colaborativa:** toda la información contenida en esta aplicación web deberá ser proporcionada por los usuarios de la misma. La aplicación web actúa como plataforma para que los estudiantes aporten información organizada en un sitio web común.

- **Sencillez en su uso:** la aplicación web será utilizada por alumnos que están familiarizados con las nuevas tecnologías y por los que no lo están tanto. Tiene que ser un diseño sencillo e intuitivo que sea atractivo para los estudiantes y que incite el uso de la aplicación web.
- **Web social:** además de conseguir realizar una aplicación web colaborativa, sería interesante aportar algún valor más social para hacer más atractiva la aplicación web, que haya algún mecanismo de interacción entre los usuarios.
- **Copias de seguridad:** debido a que son los propios usuarios los que aportan toda la información, será necesario hacer copias de seguridad diarias de la base de datos para prevenir borrados de información por parte de cualquier usuario.
- **Migrar el proyecto a un servidor remoto:** esto mejoraría el rendimiento de la aplicación web y ofrecería una mejor experiencia a los usuarios.

Capítulo 3

Estado del arte

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

3.1. Python

Python¹ es un lenguaje de programación de alto nivel con una sintaxis muy sencilla que apuesta por la simplicidad y rapidez de desarrollo. Es un lenguaje interpretado ya que no requiere que se compile el código fuente para poder ejecutarlo, algo que favorece la rapidez de desarrollo a costa de una menor velocidad de ejecución.

Su sintaxis sencilla y las normas de tabulación hacen que el código generado sea muy legible, beneficiando la lectura del código por parte del programador o por parte de cualquier otra persona.

Es un lenguaje multiparadigma, ya que permite programación orientada a objetos, programación imperativa y programación funcional.

3.1.1. Django

Django² es un framework muy utilizado para el desarrollo web y de código abierto, escrito en *Python*. Inicialmente fue desarrollado para gestionar la aplicación web de noticias de la

¹<https://www.python.org/>

²<https://www.djangoproject.com/>

World Company de Lawrance, y más tarde se liberó bajo licencia BSD.

Se basa en el paradigma MVC (Modelo-Vista-Controlador). Consiste en separar las diferentes partes del sitio: los datos de la aplicación (Modelo), la interfaz de usuario (Vista) y la lógica (Controlador). Esto es muy útil para la creación y desarrollo de aplicaciones web complejas de una manera rápida y fácil.

Por otro lado, una herramienta muy útil que ofrece **Django** es la *API Rest Framework*³. Se trata de una herramienta muy potente y flexible que facilita mucho el desarrollo del *backend*, ya que ofrece un acceso rápido a los objetos de la base de datos.

3.2. Tecnologías web

En esta sección se describen las tecnologías utilizadas para el desarrollo *frontend* de la aplicación.

3.2.1. HTML5

HTML5 [2] es la quinta versión del lenguaje de marcado *HTML*. Se utiliza para estructurar y representar el contenido de una página web. Con **HTML5**, los navegadores como *Firefox*, *Chrome*, *Safari* e incluso *Internet Explorer* pueden saber cómo mostrar una determinada página web, saber dónde están los elementos, donde colocar las imágenes o dónde el texto. La diferencia principal de **HTML5** con sus versiones anteriores es el nivel de sofisticación del código.

Una de las mejoras que proporciona **HTML5** es la posibilidad de visualizar el contenido multimedia en internet incluso sin estar conectado a la red, evitando el uso de *Flash*. Para ello se cuenta con nuevas etiquetas como `<canvas>`, `<video>` y `<audio>`, que permiten al usuario consumir videos y canciones sin necesidad de instalar nada de forma adicional. Otras nuevas etiquetas importantes de **HTML5** son `<header>`, `<footer>`, `<nav>`, `<section>` y `<article>`. Además, se añade la funcionalidad Drag & Drop, útil para arrastrar diferentes objetos.

³<http://www.django-rest-framework.org/>

3.2.2. CSS3

Las hojas de estilo en cascada *CSS* son un mecanismo que describe cómo se va a mostrar un documento en la pantalla. Se utiliza para representar de manera eficiente los documentos *HTML* y *XML*, separando el contenido de su presentación.

En 1995, el W3C apostó por desarrollar el estándar *CSS*, y en 1996 publicó la primera versión. Algunos de los estilos que permite modificar estas hojas de estilo son colores, fuentes de texto, tamaños de elementos o de la fuente y posicionamiento.

CSS evoluciona en **CSS3** [3] para ofrecer una gran variedad de opciones importantes para las necesidades del diseño web actual. Desde opciones de sombreado y redondeado y efectos en textos, hasta funciones más avanzadas de movimiento y transformación.

3.2.3. Bootstrap

Bootstrap⁴ [4] es el *framework HTML, CSS y JavaScript* más popular para el desarrollo web, gracias a la facilidad y rapidez que con la que el usuario puede desarrollar. Fue desarrollado inicialmente por *Twitter* en el año 2011 bajo licencia MIT. Es una herramienta que crea *interfaces* limpias y adaptables a todo tipo de pantallas y dispositivos de diferentes tamaños, gracias a su desarrollo *responsive*.

Bootstrap pone al servicio de los desarrolladores estilos *CSS* y *plugins JavaScript* para la creación de los diferentes elementos que pueden alojarse en los documentos *HTML*, como pueden ser tablas, formularios, botones, *breadcrumbs*, etc.

3.2.4. JavaScript

JavaScript [5] es un lenguaje de programación que se utiliza para crear páginas web dinámicas, es decir, aquellas que incorporan diferentes efectos de textos o animaciones, acciones que se activan al pulsar un botón o ventanas emergentes, y en las que se puede interactuar con otros

⁴<http://getbootstrap.com/>

usuarios. Es una implementación por parte de *Netscape* del estándar *ECMAScript*.

JavaScript comenzó a desarrollarse en los años 90 por la necesidad de los creadores de sitios web de crear páginas en las que usuarios pudieran interactuar entre ellos, y hasta entonces el *HTML* solo permitía crear páginas estáticas donde mostrar textos con estilos *CSS*. No requiere compilación ya que es ejecutado en el lado del cliente, y el navegador es el encargado de interpretar el código.

3.2.4.1. Angular JS

Angular JS⁵ [6] [7] es un *framework* de *JavaScript*, un conjunto de librerías de código abierto que permite hacer páginas web avanzadas del lado del cliente de una manera fácil.

Con este *framework* se pretende mejorar el rendimiento de la aplicación. Una parte de la lógica de maquetado y de presentación de la información en la página web de traslada del servidor hacia los clientes. Así, el servidor se descarga de trabajo y lo comparte con los clientes, mejorando el rendimiento de la aplicación.

3.3. Alojamiento web

3.3.1. Github

Git⁶ [8] es un sistema de control de versiones de aplicaciones diseñado para manejar todo tipo de proyectos, grandes y pequeños, con gran rapidez y eficacia.

El control de versiones es algo fundamental para la administración de un proyecto de desarrollo de software. Es muy importante para el trabajo en equipo, pero también es muy útil para el trabajo individual. Surge la necesidad de gestionar los cambios entre distintas versiones de un mismo código. Tarde o temprano, un programador se ha visto en la necesidad de tener dos o más copias de un mismo archivo, para no perder su estado anterior cuando se van a introducir diversos cambios. Y esto lo soluciona **Git**, guardando las diferentes versiones de los archivos,

⁵<https://angularjs.org/>

⁶<https://git-scm.com/>

y permitiendo trabajar en diferentes ramas a la vez.

En este trabajo se ha elegido **Github**⁷ como plataforma de alojamiento del código utilizando el sistema de control de versiones **Git**. Es un software que proporciona herramientas muy útiles tanto para el trabajo en equipo como para el trabajo individual.

3.3.2. OVH

OVH⁸ es el tercer proveedor de *web hosting* (alojamiento web) más utilizado del mundo. El alojamiento web es un servicio que permite a un usuario almacenar cualquier contenido para que sea accesible vía web. Para ello, debe tener asociado un dominio a través del cual se pueda visualizar el contenido alojado en el proveedor.

OVH ofrece servidores dedicados, alojamiento compartido, registro de dominios y servicios de almacenamiento en la nube. Ofrece estos servicios a precios bastante económicos, dependiendo de la necesidad del cliente.

⁷<https://github.com/>

⁸<https://www.ovh.es/>

Capítulo 4

Diseño e implementación

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

La arquitectura de la aplicación está diferenciada entre el lado del cliente y el lado del servidor, pero interactúan entre ellos. El cliente está desarrollado con *HTML5*, *CSS3* y *Javascript* mientras que el servidor se desarrolla con *Python Django*.

4.1. Arquitectura general

El proyecto consta de múltiples ficheros y directorios, representados en la Figura 4.1.

Los ficheros y directorios más destacados de un proyecto de *Django* son:

- **urls.py:** es la interfaz entre el cliente y el servidor. Maneja las peticiones HTTP recibidas y las envía a la función contenida en *views.py* correspondiente.
- **views.py:** contiene funciones que reciben una petición HTTP y devuelven una respuesta HTTP. En este proyecto sólo se reciben peticiones GET, ya que las peticiones POST se gestionan a través de la API.
- **models.py:** determina la estructura de datos. Se detalla en la sección 4.2.1
- **db.sqlite3:** es la base de datos del proyecto.
- **templates/:** contiene las plantillas *HTML* que definen la interfaz de usuario.

- **static/:** contiene ficheros estáticos *Javascript* que añaden funcionalidad en el lado del cliente y ficheros *CSS* que definen el diseño de las plantillas. Están incluidas las librerías de *Bootstrap* y *Angular JS*.

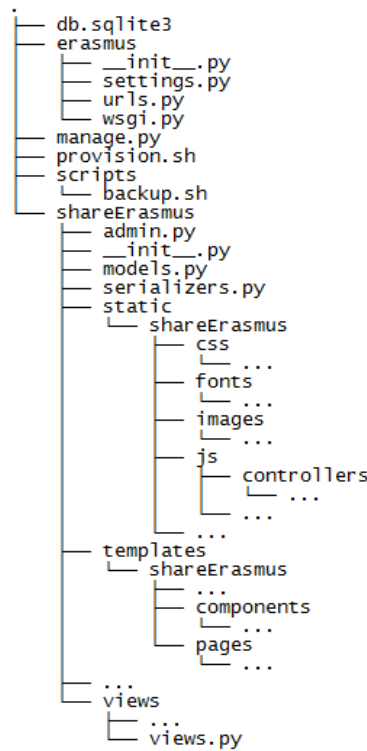


Figura 4.1: Esquema de ficheros del proyecto

Como se ha comentado anteriormente, cuando el servidor recibe una petición HTTP por parte del cliente, dicha petición es manejada desde el fichero *urls.py*. Tras encontrar la url en este fichero, se hace ejecutar la función correspondiente del fichero *views.py*. La función trata la petición GET y responde con una respuesta HTTP que devuelve una plantilla *HTML* al cliente.

4.2. Diseño e implementación del servidor

4.2.1. Modelos de la base de datos

La aplicación requiere de una base de datos donde se guarde información requerida para su correcto funcionamiento. La estructura de datos se guarda en el fichero *models.py*. Django se encarga de generar las tablas pertinentes dentro de la base de datos. La base de datos utilizada

en esta aplicación es *SQLite3*.

A continuación se enumeran y describen cada uno de los modelos creados para la aplicación web, así como los campos de cada uno de ellos:

- **Asignatura:** modelo que representa una asignatura en una universidad. El nombre de una asignatura es único solamente dentro de una universidad (otra universidad puede tener una asignatura con ese mismo nombre). Los campos del modelo son:
 - **Nombre:** nombre de la asignatura.
 - **Universidad:** referencia a la universidad donde se imparte la asignatura.
 - **Descripción:** descripción de la asignatura, donde se incluye la dificultad, el modelo de evaluación, tipo de exámenes, de prácticas, etc.
 - **Convalidaciones:** información relativa a las asignaturas que se convalidan.
 - **Créditos ECTS:** número de créditos ECTS de la asignatura.
- **Universidad:** modelo que representa una universidad. Los campos del modelo son:
 - **Nombre:** nombre de la universidad.
 - **Ciudad:** referencia a la ciudad a la que pertenece la universidad.
 - **Descripción:** descripción general de la universidad, donde pueden describirse los edificios universitarios tales como la biblioteca, la cafetería, los aularios, los laboratorios, etc.
 - **Contactos:** comentarios que aporten información para poder contactar con la universidad de destino.
 - **Latitud:** coordenada de la latitud en la que se encuentra la universidad.
 - **Longitud:** coordenada de la longitud en la que se encuentra la universidad.
- **Ciudad:** modelo que representa una ciudad. En él aparecen distintos campos en los que se guarda información de la ciudad dividida en diferentes categorías. Los campos del modelo son:
 - **Nombre:** nombre de la ciudad.

- **País:** referencia al país al que pertenece la ciudad.
 - **Descripción:** descripción general de la ciudad.
 - **Alojamiento:** información relativa a dónde alojarse en la ciudad.
 - **Transporte:** información relativa al transporte en la ciudad.
 - **Precios:** información relativa a los precios generales de la vida en la ciudad.
 - **Telefonía móvil:** información relativa a las compañías telefónicas recomendadas.
 - **Clima:** información relativa al clima en la ciudad.
 - **Vida estudiantil:** información relativa al estilo de vida en la ciudad.
 - **Vida nocturna:** información relativa a lugares de salida nocturna recomendados.
 - **Cuenta bancaria:** información relativa a los bancos recomendados para abrir una cuenta, sacar dinero, etc.
 - **Restaurantes:** información relativa a restaurantes o bares donde comer.
 - **Compras:** información relativa a los lugares para comprar en la ciudad.
 - **Cultura:** información relativa a lugares y monumentos de la ciudad.
 - **Turismo:** información relativa a otras ciudades cercanas recomendadas para visitar.
 - **Otra información:** otra información de la ciudad que no encaje en ninguno de los campos anteriores.
- **País:** modelo que representa un país. El campo del modelo es:
 - **Nombre:** nombre del país.
 - **Perfil de usuario:** modelo que representa a un usuario. Es un modelo que hereda los campos del modelo User de *Django*. Los campos importantes heredados son:
 - **Nombre de usuario:** nombre de usuario. Debe ser único. Será el identificador utilizado para iniciar sesión.
 - **Correo electrónico:** correo electrónico del usuario. Debe ser único.
 - **Contraseña:** contraseña de la cuenta del usuario.
 - **Nombre:** nombre del usuario.

- **Apellidos:** apellidos del usuario.

Además, se han añadido los siguientes campos:

- **Correo electrónico público:** boolean que indica si se muestra o no el correo electrónico a los visitantes de la aplicación web.
- **Asignaturas:** referencia a las asignaturas cursadas por el usuario.
- **Comentario:** modelo que representa un comentario realizado sobre una universidad o sobre una asignatura. Los campos del modelo son:
 - **Usuario:** referencia al perfil de usuario que realiza el comentario.
 - **Cuerpo:** contenido del comentario.
 - **Fecha:** día y hora en el que se realizó el comentario.
 - **Universidad:** referencia a la universidad donde se ha escrito el comentario. Es posible que este campo no contenga ninguna referencia.
 - **Asignatura:** referencia a la asignatura donde se ha escrito el comentario. Es posible que este campo no tenga ninguna referencia.

4.2.2. API

Se ha creado una API REST que permite el acceso a los datos de la base de datos a través de una interfaz web. De esta manera se consigue reducir en parte la carga computacional del servidor, ya que cada vez que se carga una página, el servidor no se encarga de insertar los datos de la base de datos en la plantilla, sino que esto se realiza desde el lado del cliente con funciones *Javascript* que realizan peticiones a la API.

La API también es la encargada de gestionar las peticiones POST que realizan los clientes al enviar un formulario desde las páginas de configuración que se describen en la sección 4.3.2. Las peticiones POST se gestionan desde el fichero *api.py*.

Los path de la API son:

- **api/1.0/countries/:** contiene la información referida a los países.

- **api/1.0/countries/(id-country)/:** contiene la información referida al país *id-country*.
- **api/1.0/cities/:** contiene la información referida a las ciudades.
- **api/1.0/cities/(id-city)/:** contiene la información referida a la ciudad *id-city*.
- **api/1.0/universities/:** contiene la información referida a las universidades.
- **api/1.0/universities/(id-university)/:** contiene la información referida a la universidad *id-university*.
- **api/1.0/users/:** contiene la información referida a los usuarios.
- **api/1.0/users/(id-user)/:** contiene la información referida al usuario *id-user*.
- **api/1.0/subjects/:** contiene la información referida a las asignaturas.
- **api/1.0/subjects/(id-subject)/:** contiene la información referida a la asignatura *id-subject*.
- **api/1.0/comments/:** contiene la información referida a los comentarios.
- **api/1.0/comments/(id-comment)/:** contiene la información referida al comentario *id-comment*.

4.2.3. Backups

Un elemento importante que no puede faltar en una aplicación de estas características es el *backup*. Tratándose de una aplicación web **colaborativa**, es necesario realizar múltiples copias de seguridad de la base de datos ya que hay mucha información que puede ser modificada por cualquier usuario, pudiendo hacer borrados menores o incluso masivos. De esta manera se pretende evitar la pérdida de información que pudieran ocasionar usuarios maliciosos, o simplemente despistados.

Para hacer copias de la base de datos automáticamente se utiliza una herramienta de *Linux* llamada *cron*, la cual permite ejecutar tareas repetitivas periódicamente. En este caso el *script* que se quiere ejecutar es *backup.sh*. Para ello [9] es necesario editar el fichero */etc/crontab* (Ver Figura 4.2) y añadir al final la siguiente línea:

0 3 * * * root sh /tfg/tfg/scripts/backup.sh. El archivo de *crontab* tiene la siguiente estructura de izquierda a derecha: Minutos (rango de 0-59), Horas (0-23), Día del mes (1-31), Mes (1-12), Día de la semana (0-6 siendo 0=Domingo), Path completo al *script* o programa que queramos ejecutar.

```

GNU nano 2.2.6      File: /etc/crontab      Modified
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --repo$
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --repo$
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --repo$
#
0 3 * * * root sh /tfg/tfg/scripts/backup.sh

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell

```

Figura 4.2: Archivo /etc/crontab

En el *script* mencionado anteriormente se ejecutan los comandos necesarios para hacer copias de la base de datos. Para no saturar el servidor, se realiza una copia cada día, y se almacena durante una semana. Así se evita que una gran cantidad de copias de la base de datos pueda ocupar demasiado espacio en el servidor pudiendo colapsarlo. Por lo tanto siempre hay 7 copias correspondientes a los últimos 7 días, y en el momento en el que se detectara algún borrado se sustituiría la base de datos por una de las guardadas anteriormente.

4.3. Diseño e implementación del cliente

El cliente puede acceder a la interfaz de la aplicación a través de un navegador web. Como se ha comentado en la sección 3.2, para el desarrollo del lado del cliente se han utilizado las tecnologías *HTML5*, *CSS3* y *Javascript*. Las dos primeras son necesarias para definir cómo se visualizan y con qué estilos y diseño cada uno de los elementos visibles en la aplicación, co-

mo puede ser la barra del menú superior, el listado de las universidades, los comentarios, etc. *Javascript*, y más concretamente el *framework Angular JS*, se utiliza para gestionar el flujo de información de la parte del cliente. Esto es posible con la utilización de los controladores de *Angular JS*. Así es posible incluir en las vistas de la aplicación web los datos obtenidos de la base de datos a través de peticiones a la API.

En cuanto a la interfaz, se ha utilizado la plantilla *Margo* [10]. La plantilla se ha adaptado a las necesidades propias de la aplicación y se han creado nuevos elementos y estilos para hacerla más atractiva. Entre las características que se han añadido ha sido la creación de usuarios, algo que no contempla la plantilla original.

Interfaz para dispositivos móviles

La interfaz de usuario utiliza la técnica *Responsive Design*, lo que permite que se adapte a cualquier tipo de pantalla y dispositivos de cualquier tamaño.

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 son públicas y accesibles para todo tipo de usuarios, tanto para usuarios registrados como para los no registrados.

En todas las páginas aparece una barra superior, a través de la cual se puede acceder de manera sencilla a la página de inicio, la página de universidades, la página de ayuda y a la página de contacto. Además, si el usuario no está registrado, aparece un enlace hacia la página de inicio de sesión y registro, y si el usuario está registrado, los enlaces se corresponden con la página de perfil del usuario, las distintas páginas de configuración (que se detallan en la sección 4.3.2) y un enlace para cerrar la sesión.

Página de inicio

En la vista de inicio¹ se hace un resumen de las características que tiene la aplicación web para hacer más fácil la navegación a través de ella (ver Figura 4.3).

¹url relativa: /



Figura 4.3: Página de inicio

Página de inicio de sesión y registro

La vista de inicio de sesión y registro² está dividida en dos partes (ver Figura 4.4). En el lado derecho aparece un formulario que permite registrar a un nuevo usuario de una manera rápida, y en el lado izquierdo está el formulario para iniciar sesión con una cuenta registrada anteriormente.

²url relativa: /sign

The screenshot shows the 'SHARE ERASMUS' web application. At the top, there is a header with the site name, a navigation menu with links for 'Inicio', 'Universidades', 'Contacto', and 'Iniciar Sesión/Registrarse' (which is highlighted), and a Twitter icon. Below the header, the page is split into two main sections: 'Iniciar sesión' (Login) on the left and 'Registrarse' (Register) on the right. The login section has fields for 'Nombre de usuario' and 'Contraseña', followed by an 'Entrar' button. The registration section has fields for 'Correo electrónico', 'Nombre de usuario', 'Contraseña', and 'Confirmar contraseña', followed by a checkbox for accepting the 'política de privacidad' and a 'Crear cuenta' button. A footer at the bottom contains copyright information and links to 'PRIVACY POLICY' and 'CONTACT'.

Figura 4.4: Página de inicio de sesión y registro

En el formulario de registro se requiere que el usuario introduzca su cuenta de correo electrónico, un nombre de usuario y la contraseña. Además, se pide que se lea y acepte la política de privacidad. El correo electrónico no es público para el resto de personas a no ser que el propio usuario así lo indique. El correo electrónico puede hacerse público en la página de configuración de la cuenta.

La autenticación en la aplicación se realiza a través del formulario de inicio de sesión. El identificador para iniciar sesión es el nombre de usuario, que irá acompañado de su correspondiente contraseña.

Página de universidades

En esta vista³ se muestra un listado de todas las universidades añadidas por los usuarios registrados en la aplicación (Ver Figura 4.5).

Lo primero que cabe destacar de esta vista es el buscador superior. Se trata de una búsqueda predictiva. Al introducir un texto, aparecen todos los resultados de universidades que contienen la cadena de texto introducida. Esto se realiza con la librería *angucomplete-alt* [11] de *Angular JS*, cuya función es buscar a través de la API las coincidencias entre el texto introducido en el buscador y los nombres de las universidades guardadas en la base de datos. Debajo de este

³url relativa: /universities

buscador se muestra un enlace a través del cual, sólo si el usuario está registrado, se accede a la página de configuración donde se puede crear una nueva universidad.

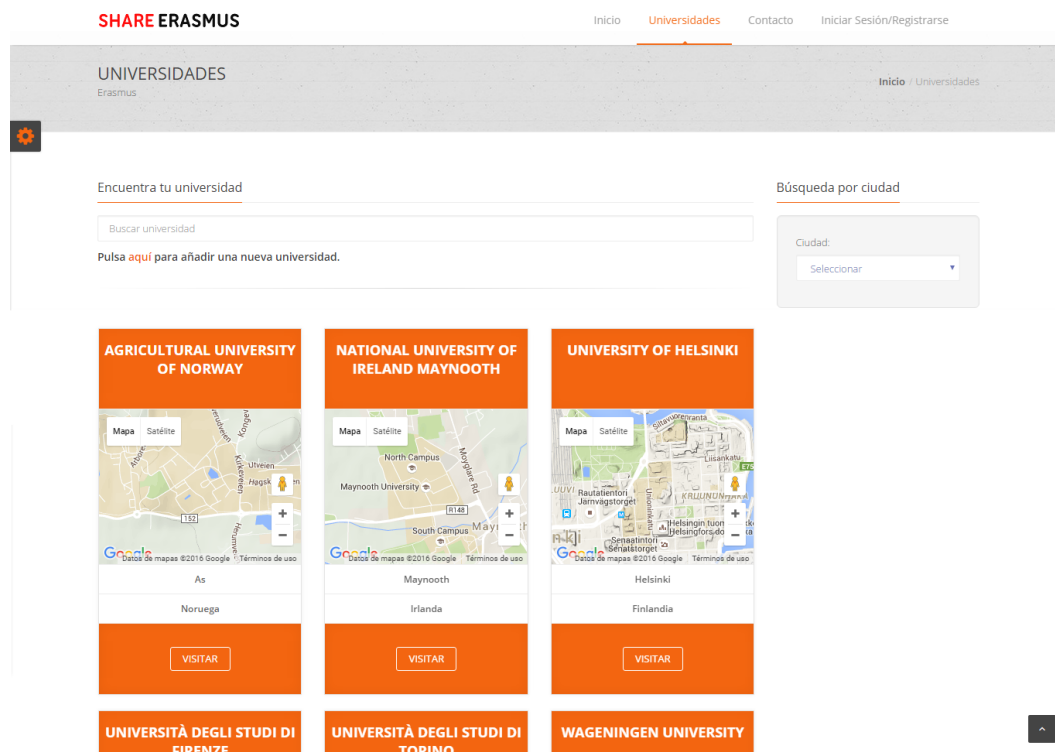


Figura 4.5: Página de universidades

En la parte central de la vista se muestra una lista de todas las universidades, distribuidas en filas de tres en tres y ordenadas alfabéticamente. De cada universidad se muestra su nombre, un mapa de Google Maps⁴ que muestra su localización, el nombre de la ciudad de la universidad y su país. Además a través del botón *VISITAR* se accede a la página de esa universidad.

En la parte de la derecha se muestra un filtro que permite mostrar únicamente todas las universidades que se localicen en la ciudad seleccionada.

Tras haber realizado un filtrado, debajo del buscador superior aparece un botón que permite volver a mostrar el listado de todas las universidades.

Página de una universidad

⁴<https://developers.google.com/maps/documentation/geocoding/intro>

En esta vista⁵ se muestra información relativa a la universidad y a su ciudad (Ver Figura 4.6). La información relativa a la universidad la pueden añadir los usuarios desde la página de configuración de mis universidades, y la relativa a la ciudad desde la página de configuración de mis ciudades.

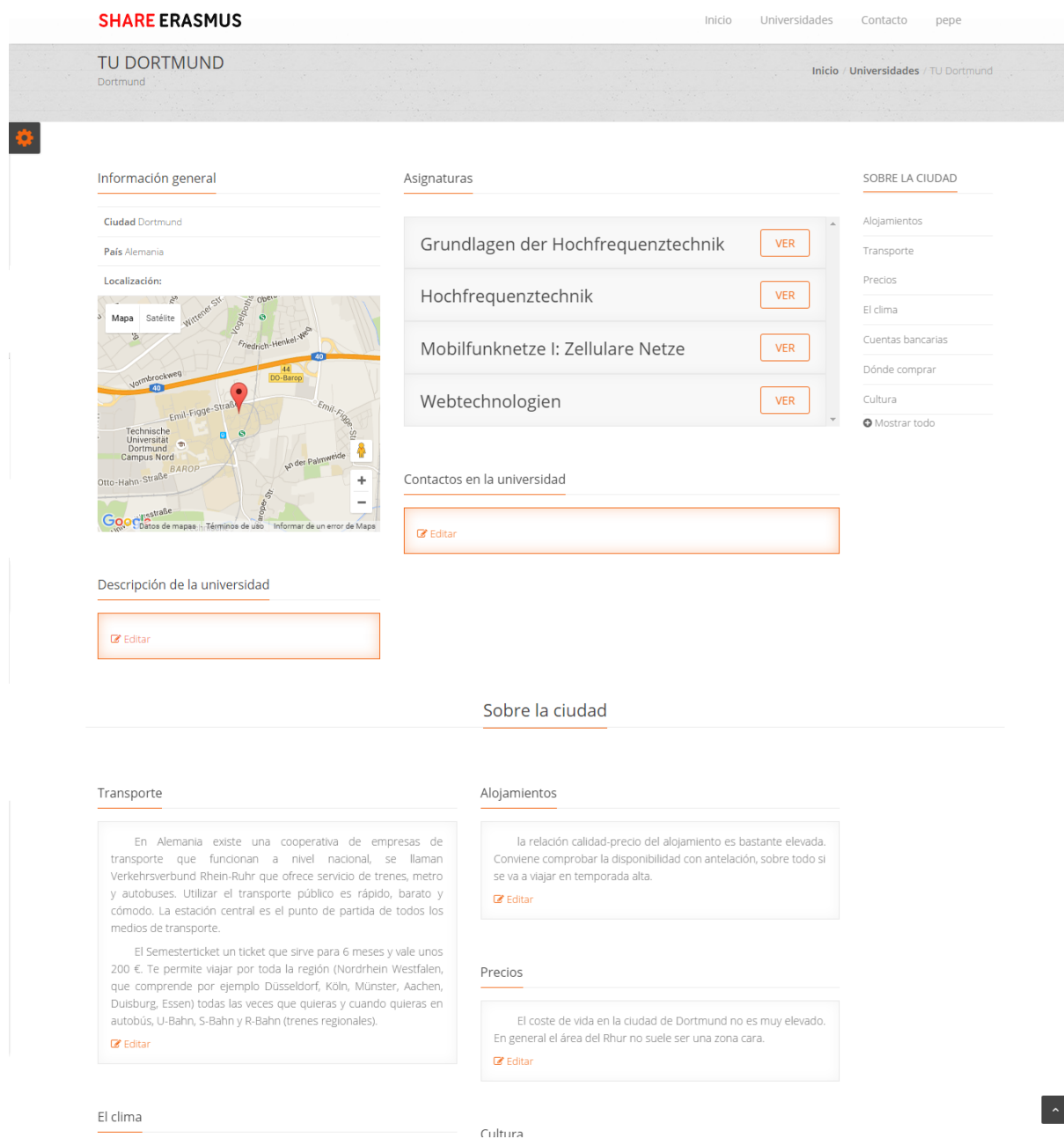


Figura 4.6: Página de una universidad

En la parte superior izquierda se visualiza la información general de la universidad, como

⁵url relativa:/universities/(id-university)

es el nombre de su ciudad y su país, además de mostrar un mapa de Google con su localización.

En la parte superior central se muestra un listado de todas las asignaturas que se imparten en la universidad. A través del botón *VER* de cada una de ellas se accede a la página de esa asignatura.

Justo debajo de la información general de la universidad y del listado de asignaturas se muestra distinta información relativa a la universidad.

En el lateral derecho de la vista se muestra un listado de las distintas categorías con información relativa a la ciudad. Al pulsar sobre cualquiera de ellas se accede directamente a visualizar la información sobre esa categoría. Al final de la listas de categorías hay un botón que al pulsarlo muestra la lista completa de categorías, tanto las que tienen información como las que no. Toda la información relativa a la ciudad se muestra en esta vista debajo de la información relativa a la universidad.

Por último, debajo de toda la información de la ciudad aparecen comentarios que pueden realizar los usuarios. Solo pueden comentar usuarios registrados, aunque los comentarios son visibles por cualquier visitante de la página.

Página de una asignatura

En esta vista⁶ (Ver Figura 4.9) se muestra información relativa a una asignatura. Esta información puede ser añadida por los usuarios desde la página de configuración de mis asignaturas.

A la izquierda se muestra el número de créditos ETCS de los que consta la asignatura, una descripción e información relativa a las convalidaciones de las asignaturas.

A la derecha aparece un listado de los usuarios que han seleccionado la asignatura desde la página de configuración de mis universidades. Al pulsar sobre el nombre de un alumno se accede a la página de perfil de ese usuario.

⁶url relativa:/universities/(id-university)/(id-subject)

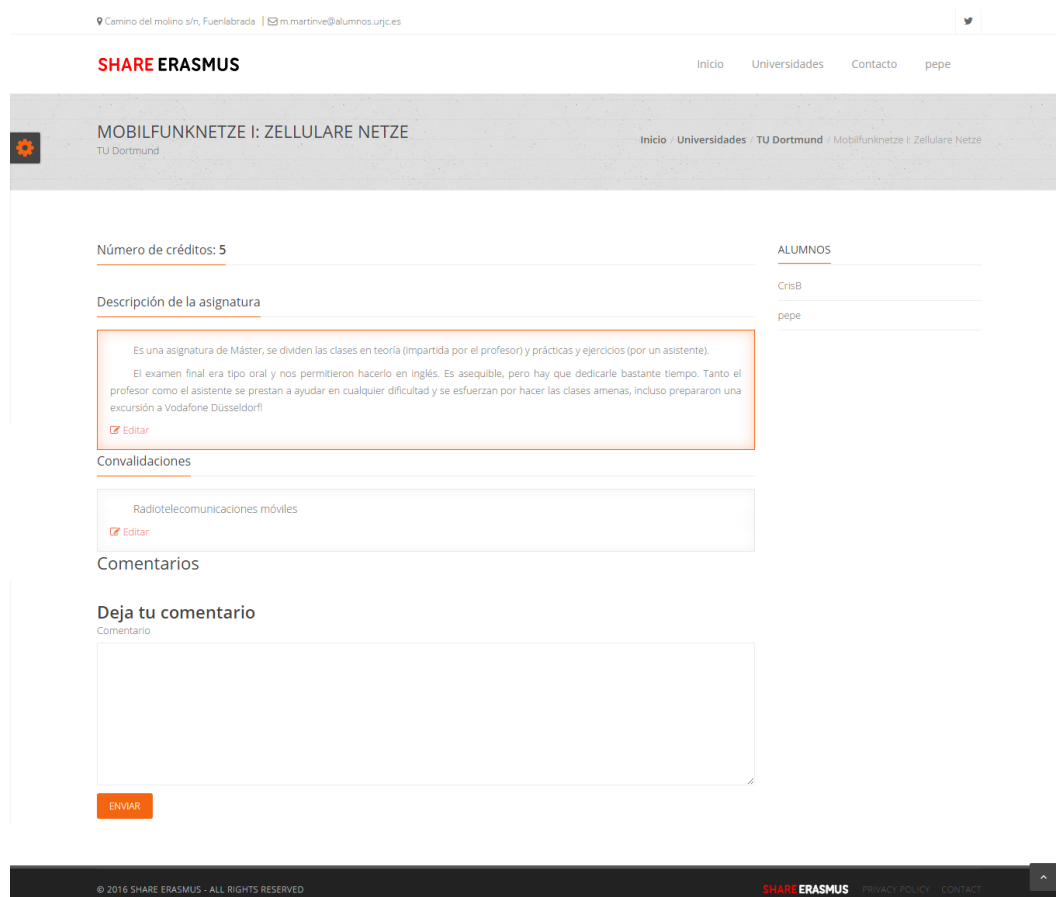


Figura 4.7: Página de una asignatura

Por último, al igual que en la página de una asignatura, debajo de la información de la asignatura aparecen comentarios realizados por los usuarios.

Página de perfil de un usuario

En esta vista⁷ se muestra información sobre un usuario (Ver Figura 4.8).

Se muestra una barra en la parte superior de la página donde aparece el nombre del usuario y un enlace a la página de inicio. Más abajo se muestra una imagen de perfil. Además, si el usuario permite que se muestre su correo electrónico, aparecerá junto al nombre completo del

⁷url relativa: /users/ (username)

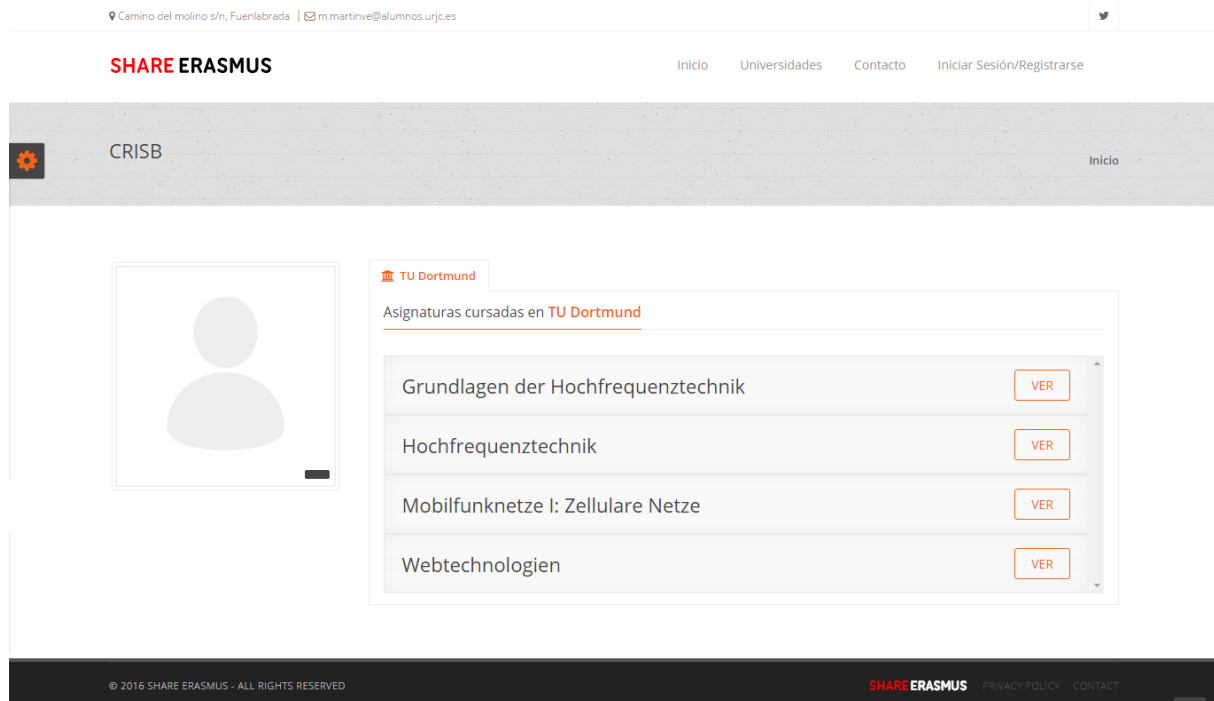


Figura 4.8: Página de un usuario

usuario (si ha proporcionado la información en la página de configuración de la cuenta).

A la derecha se muestran las universidades donde el usuario ha indicado que ha estudiado o tiene intención de estudiar. Al pulsar sobre el nombre de la universidad aparece una lista con las asignaturas cursadas o por cursar del usuario. Al pulsar en el botón *VER* de una asignatura se accede a la página de esa asignatura.

Página de contacto

A través de esta vista⁸ se puede contactar con el administrador de la aplicación para cualquier duda o sugerencia (Ver Figura 4.9).

⁸url relativa: /contact

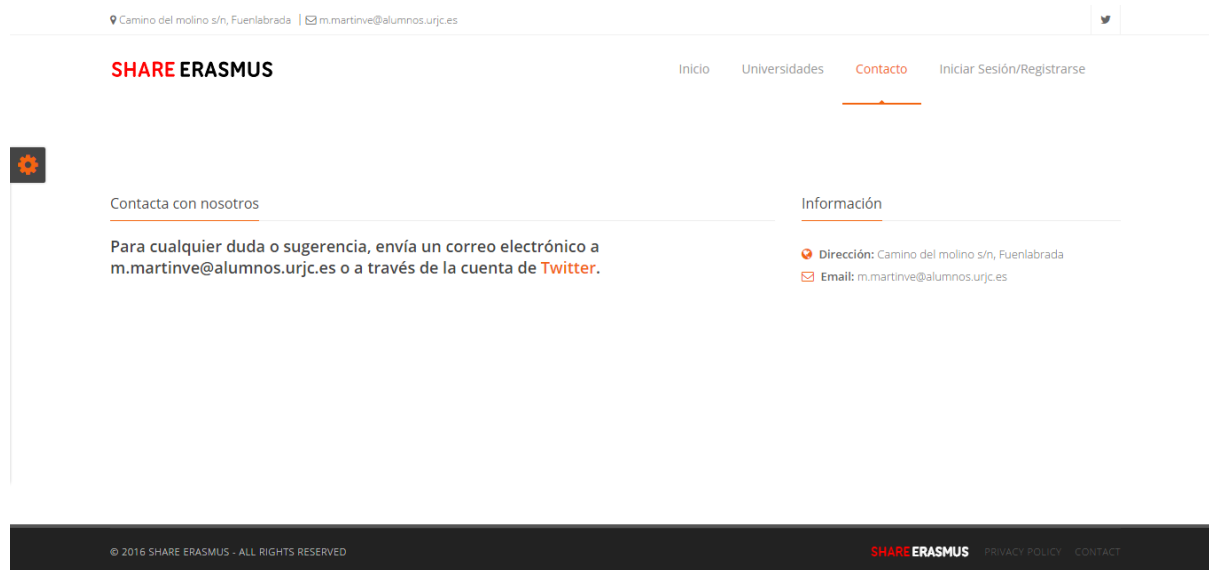


Figura 4.9: Página de contacto

Página de política de privacidad

En esta vista⁹ se muestra información relativa a la información que se recopila y puede ser visible de los usuarios en esta aplicación (Ver Figura 4.10).

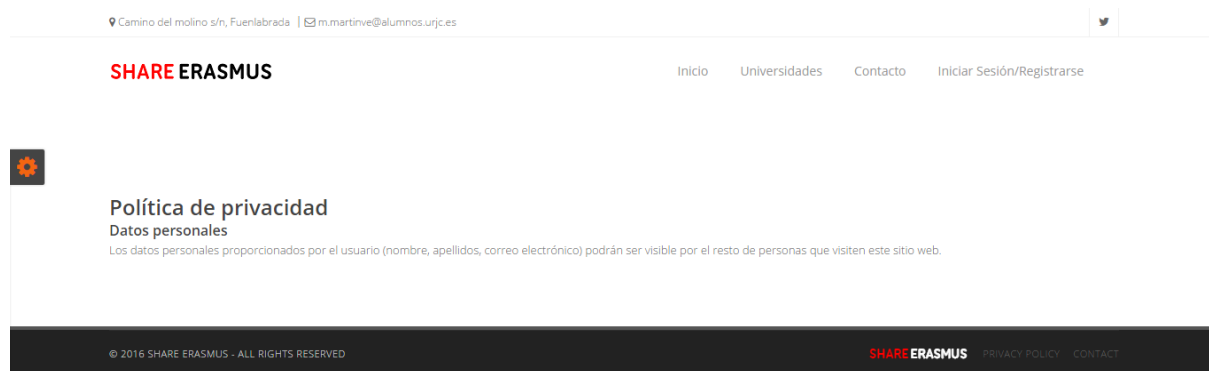


Figura 4.10: Página de política de privacidad

4.3.2. Vistas de la configuración

A continuación se describen las vistas de las páginas que son accesibles únicamente por usuarios registrados en la aplicación. Se puede acceder desde los enlaces que aparecen a la derecha en la barra superior. Desde estas vistas se completa tanto información personal del usuario

⁹url relativa: /privacy_policy

como información relevante de las universidades, ciudades o asignaturas cursadas por el usuario.

Todas las vistas de configuración tienen un menú en el lateral izquierdo desde donde se puede acceder a los diferentes sitios de configuración.

Página de configuración de mi cuenta

En esta vista¹⁰ aparece un formulario donde se pide al usuario (si lo desea) completar algunos datos personales (Ver Figura 4.11).

SHARE ERASMUS Inicio Universidades Datos actualizados con éxito

CONFIGURACIÓN

Cuenta

Contraseña

Mis universidades

Mis asignaturas

Mis ciudades

Cambia tus configuraciones básicas de tu cuenta

Nombre de usuario pepe

Nombre Pepe

Apellidos Martínez Martín

Correo electrónico pepe@hotmail.com

☒ Permiso que mi correo electrónico sea visible por todo el mundo.

Actualizar

© 2016 SHARE ERASMUS - ALL RIGHTS RESERVED SHARE ERASMUS PRIVACY POLICY CONTACT

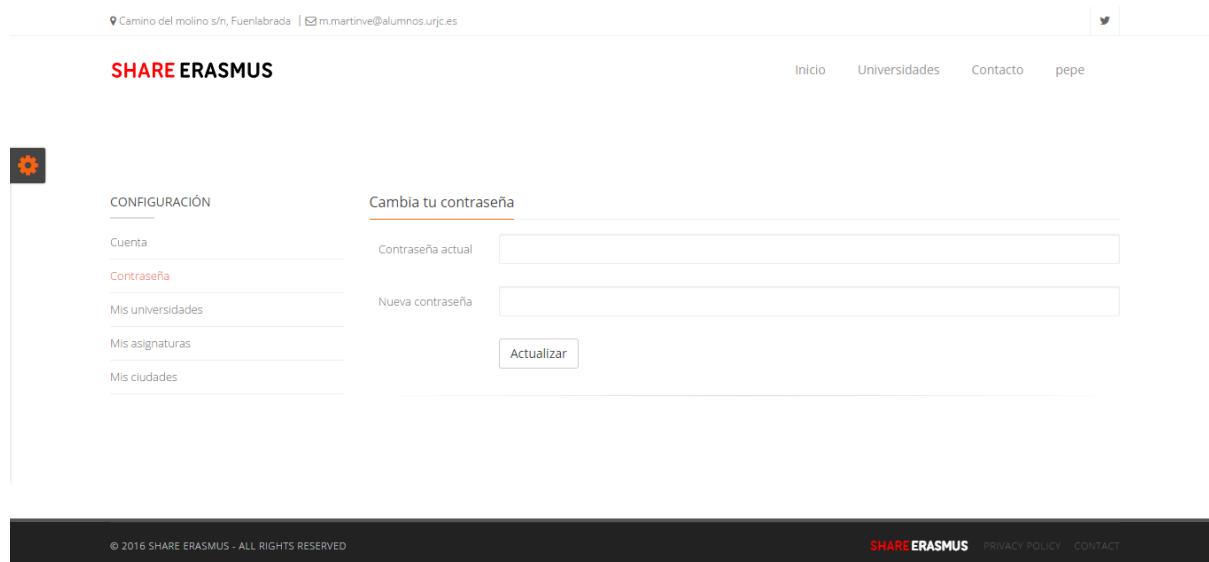
Figura 4.11: Página de configuración de la cuenta

Puede modificarse el nombre de usuario y el correo electrónico con los que el usuario se registró y añadir información como el nombre y los apellidos. Además aparece un *checkbox* que puede marcarse si el usuario quiere que el correo electrónico pueda ser visible por cualquier persona que visite su página de perfil.

Página de cambio de contraseña

¹⁰url relativa: /settings/account

En esta vista¹¹ aparece un formulario donde se puede cambiar la contraseña de la cuenta (Ver Figura 4.12). En el formulario se deben introducir la contraseña actual y la nueva contraseña.



The screenshot shows the 'SHARE ERASMUS' website interface. At the top, there is a header with the site name and navigation links: 'Inicio', 'Universidades', 'Contacto', and 'pepe'. Below the header, a sidebar on the left contains a settings icon and a list of menu items: 'CONFIGURACIÓN', 'Cuenta', 'Contraseña', 'Mis universidades', 'Mis asignaturas', and 'Mis ciudades'. The main content area is titled 'Cambia tu contraseña' and contains two input fields labeled 'Contraseña actual' and 'Nueva contraseña', followed by an 'Actualizar' button. The footer at the bottom includes copyright information '© 2016 SHARE ERASMUS - ALL RIGHTS RESERVED' and links to 'SHARE ERASMUS', 'PRIVACY POLICY', and 'CONTACT'.

Figura 4.12: Página de cambio de contraseña

Página de configuración de mis universidades

Desde esta vista¹² se pueden hacer dos funciones: buscar universidad y crear universidad. Lo primero que debe hacer un usuario cuando quiera registrarse en una universidad y en sus asignaturas cursadas o por cursar, es buscar la universidad en cuestión por si ya hubiera sido creada anteriormente. (Ver Figura 4.13)

Al pulsar sobre *Buscar universidad* aparecen varios filtros, que hay que utilizarlos en orden. En el primer filtro se debe seleccionar el país donde se encuentra la universidad, después se selecciona la ciudad y finalmente la universidad. A continuación se muestran algunas asignaturas que se imparten en dicha universidad, y el usuario debe seleccionar las que haya cursado o esté interesado en cursar. Además, si hay alguna asignatura que no aparezca en la lista, se pueden crear un máximo de 6 asignaturas a la vez. Al pulsar en *Guardar* quedan registradas las

¹¹url relativa: /settings/password

¹²url relativa: /settings/universities

asignaturas en la cuenta del usuario.

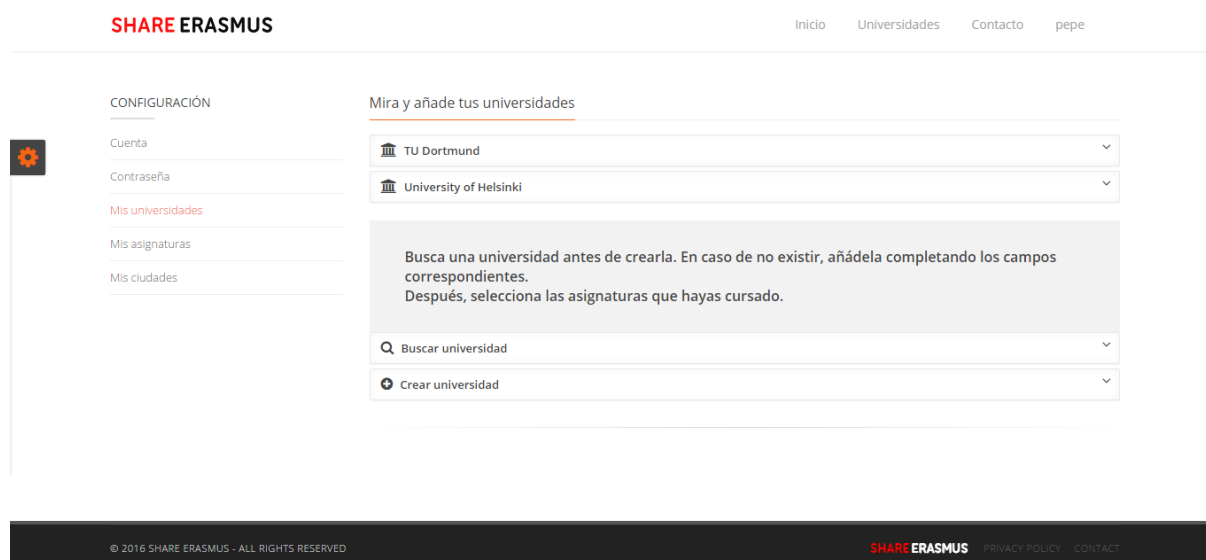


Figura 4.13: Página de configuración de mis universidades

Al recargar la página se muestra en la parte superior el nombre de la universidad. Al pulsar sobre él, aparece un formulario con varios campos de texto para completar información sobre la universidad: descripción y contactos (descritos en 4.2.1). Al ser una aplicación web colaborativa, todos los usuarios registrados en una universidad pueden modificar esa información. Se trata de complementar información que han escrito otros usuarios o añadir información nueva, tratando de que no se elimine información escrita por otros usuarios.

En caso de haber buscado una universidad y no haberla encontrado, existe la opción de crear una universidad. Al pulsar sobre *Crear universidad* aparece un formulario en el que se debe introducir el nombre de la universidad, su ciudad y su país. Tras guardarla se debe recargar la página para buscar la universidad (de la manera descrita anteriormente) y seleccionar las asignaturas que sean oportunas.

Página de configuración de mis asignaturas

En esta vista¹³ se muestra un listado de las asignaturas seleccionadas desde la vista de con-

¹³url relativa: /settings/subjects

figuración de mis universidades (Ver Figura 4.14).

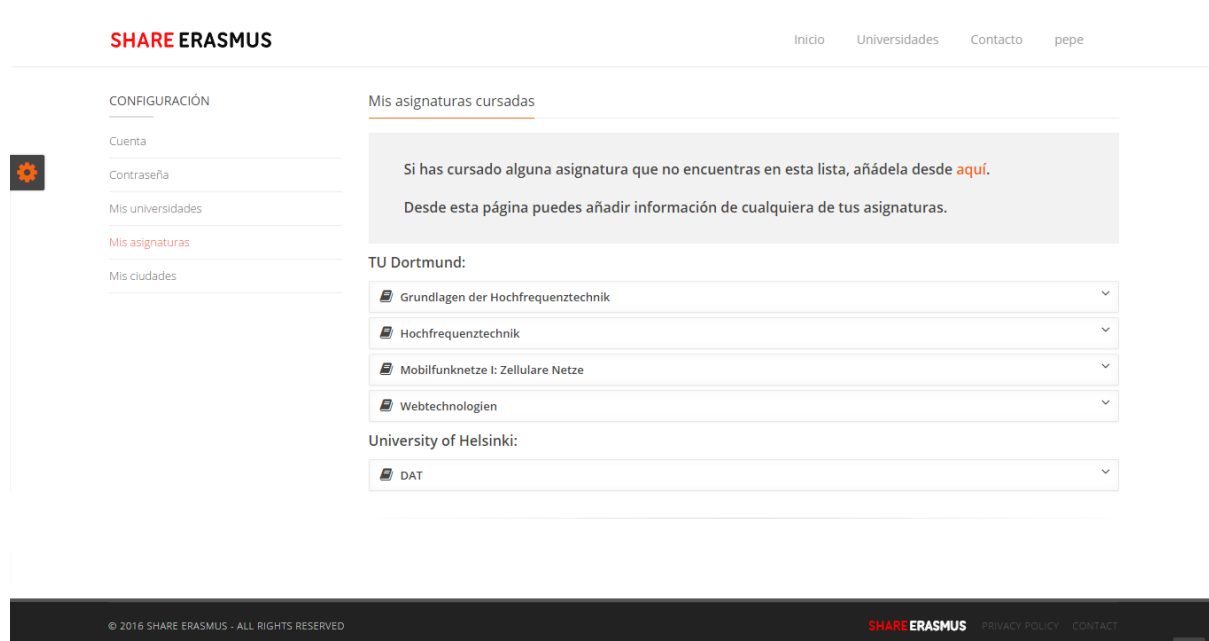


Figura 4.14: Página de configuración de mis asignaturas

Al pulsar sobre una de ellas se muestra un formulario para completar información sobre la asignatura. Esta información es la que aparece en la página de la asignatura.

La información que puede añadirse es el número de créditos ECTS que se convalidan y una descripción, en la que se podría añadir información relativa a la dificultad de la asignatura, la forma de evaluación, tipo de exámenes, prácticas, etc. Es decir, cualquier información que otro usuario quisiera conocer antes de decidirse por escoger esa asignatura para sus estudios Erasmus.

Página de configuración de mis ciudades

En esta vista¹⁴ se muestra un listado de las ciudades donde el alumno ha estudiado, en el caso de ser en más de un lugar (Ver Figura 4.15).

Al pulsar sobre el nombre de la ciudad se muestra un amplio formulario que permite com-

¹⁴url relativa: /settings/cities

pletar información relativa a la ciudad. Esta información se muestra en la páginas de las universidades que pertenezcan a la ciudad.

The screenshot shows the 'SHARE ERASMUS' web application. At the top, there is a header with the location 'Camino del molino s/n, Fuenlabrada' and email 'm.martine@alumnos.urjc.es'. The main navigation bar includes 'Inicio', 'Universidades', 'Contacto', and 'pepe'. On the left, a sidebar menu lists 'CONFIGURACIÓN' with sub-items: 'Cuenta', 'Contraseña', 'Mis universidades', 'Mis asignaturas', and 'Mis ciudades' (highlighted in red). The main content area is titled 'Añade información de las ciudades donde has estudiado'. It contains a text box with instructions: 'Aquí podrás añadir o modificar información sobre los diferentes temas que se proponen de una ciudad. Si ves que lo que quieres escribir ya está, no lo repitas, aunque puedes complementarlo con más información o reescribiéndolo, pero trata de no borrar nada escrito por otro compañero.' Below this are two dropdown menus, each with a flag icon and a downward arrow. The first dropdown is labeled 'Dortmund' and the second is labeled 'Helsinki'. The footer contains copyright information '© 2016 SHARE ERASMUS - ALL RIGHTS RESERVED' and links to 'SHARE ERASMUS', 'PRIVACY POLICY', and 'CONTACT'.

Figura 4.15: Página de configuración de mis ciudades

El formulario está dividido en diferentes categorías. No es necesario que el usuario rellene información de todas las categorías, aunque sí sería recomendable por el bien de la comunidad de la aplicación. Las diferentes categorías son: descripción de la ciudad, información sobre el alojamiento en la ciudad, el transporte, los precios, la telefonía móvil, la climatología, la vida de estudiante, la vida nocturna, bancos o cajeros, restaurantes o bares, *shopping*, monumentos o lugares turísticos, lugares que visitar fuera de la ciudad y otra información relevante que no esté representada en ninguna categoría. Todas estas categorías están descritas en 4.2.1.

4.3.3. Vista del panel de administración

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. En la Figura 4.16 se observa la vista principal del panel de administración.

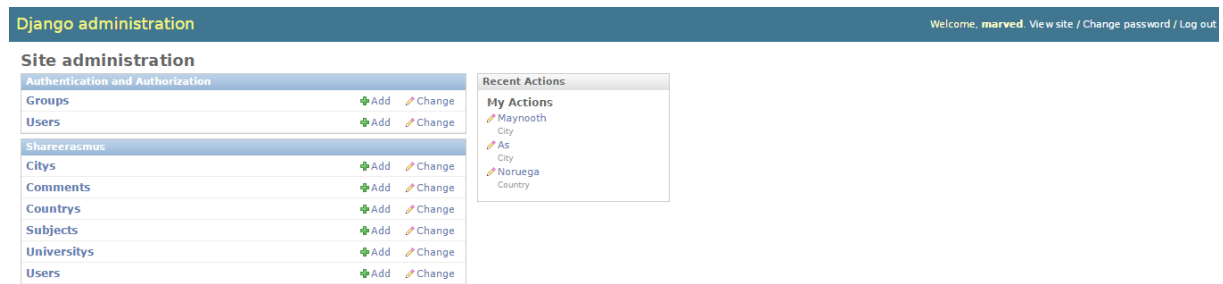


Figura 4.16: Página de administración

Capítulo 5

Resultados

A medida que se va desarrollando la aplicación web es necesario probar su funcionamiento con cada nueva funcionalidad añadida. Para ello se creó un entorno virtual de desarrollo con *Vagrant*, ya que permite instalar y configurar software en una máquina virtual que simula estar en el servidor en el que se alojará la aplicación web.

Una vez que la aplicación ya estaba bastante avanzada, se procedió a desplegarla en un servidor remoto. Se contrató el servicio de alojamiento web que ofrece OVH. De esta manera la aplicación ya podía ser utilizada por los usuarios.

Fue importante el momento en el que se hizo el despliegue en el servidor remoto ya que permitió que visitaran la web algunas personas que colaboraron con sus ideas en la mejora de la aplicación.

Todas las pruebas de funcionalidad de la aplicación se han realizado en un entorno local (preproducción), y una vez que se comprobaba que todo funcionaba correctamente se procedía a introducir los cambios en el servidor remoto (producción).

5.1. Pruebas en servidor local

Como se ha comentado anteriormente, se ha utilizado *Vagrant* para crear un entorno virtual de desarrollo. Para ello ha sido necesario instalar *VirtualBox*¹ y *Vagrant*². También ha sido necesario instalar en la máquina virtual *Django*, *Git* y los diferentes paquetes y librerías necesarias para el correcto funcionamiento de la aplicación web. Así, se ha creado el archivo de instalación *provision.sh*.

Una vez instalado todo lo necesario, para ejecutar la máquina virtual es necesario introducir en una *Shell Unix*:

```
$vagrant up
$vagrant provision
```

Para probar la aplicación web desde el navegador hay que introducir una url que contenga la dirección IP y el puerto configurados en el archivo *Vagrantfile* de esta forma: *http://IP:PUERTO/*.

En este entorno es en el que se han ido probando todas las funcionalidades nuevas que se han ido desarrollando, lo que ha permitido corregir errores y revertirlos.

5.2. Pruebas en servidor remoto

Se ha contratado un servidor VPS en la web de OVH, que ofrece un buen rendimiento y seguridad. Además ha sido necesario contratar un dominio a través del cual los usuarios pudieran acceder a la aplicación web. El dominio elegido ha sido *shareerasmus.com*.

Tras acceder mediante *ssh* a la máquina virtual contratada, lo primero que ha sido necesario hacer es clonar el repositorio *Git*³ (donde está el trabajo alojado) en la máquina virtual, para posteriormente ejecutar el archivo *provision.sh*.

¹<https://www.virtualbox.org/wiki/Downloads>

²<https://www.vagrantup.com/downloads.html>

³<https://github.com/marved/tfg>

Tras tener todo el entorno instalado, para ejecutar la aplicación se necesita ejecutar en la *Shell Unix* el comando:

```
$python manage.py runserver shareerasmus.com:80
```

Como se ha comentado anteriormente, aquí solo se despliega la aplicación cuando ha sido revisada desde el entorno local y se ha comprobado que las nuevas funcionalidades que se han ido desarrollando funcionan correctamente. Las pruebas en este entorno las realizan los usuarios informando de posibles errores o aconsejando mejoras que pudieran hacerse para mejorar la experiencia en la navegación a través de la web.

5.3. Resultado final

El resultado final es satisfactorio. Se ha conseguido desplegar la aplicación tanto en el servidor local como en uno remoto, pudiendo ser accesible para todos los usuarios que visitaran la página `shareerasmus.com`. Se han ido solventando, con mayores o con menores problemas todos los *bugs* que han ido apareciendo, y aunque sería mejorable la velocidad de navegación, se ha conseguido una aplicación web útil para la comunidad Erasmus y fácilmente navegable.

Capítulo 6

Conclusiones

6.1. Consecución de objetivos

Este trabajo pretendía crear una aplicación web colaborativa que fuera útil para futuros alumnos Erasmus. La aplicación debía proporcionar información relevante que hiciera más fácil a los estudiantes elegir el destino europeo preferido. Y la aplicación tiene la capacidad de contener toda esa información bien organizada y dividida en distintas categorías.

A continuación se enumeran los objetivos específicos que se han conseguido y los que se han quedado incompletos:

- **Estudiar las necesidades de los estudiantes:** se ha contactado con varios alumnos y alumnas que han disfrutado de una beca Erasmus, ya sean de la ETSIT o no, que han colaborado para conseguir que la temática de la información contenida en la aplicación resulte útil para los futuros estudiantes Erasmus.
- **Uso de tecnologías web avanzadas:** se han utilizado varias tecnologías web. Las que más cabe destacar son las mencionadas varias veces en anteriores capítulos, como son *HTML5*, *CSS3*, *Angular JS* y *Django*. Todas ellas han facilitado el desarrollo de esta aplicación web gracias a sus grandes funcionalidades y características, consiguiendo un buen resultado final.
- **Web colaborativa:** Toda la información que se puede encontrar en la aplicación ha sido creada por los usuarios que la han utilizado. No ha habido muchos usuarios, pero si se

consiguiera que el número se incrementara, podría ser una web muy útil para todos los futuros estudiantes Erasmus de la ETSIT.

- **Sencillez en su uso:** se ha realizado una interfaz bastante sencilla e intuitiva, para que la experiencia de navegación de los usuarios sea lo más cómoda posible. Además, en las páginas de configuración se han escrito comentarios para que los usuarios tengan claro lo que significa cada formulario y desde dónde hay que realizar cada función que se quiera hacer.
- **Web social:** se ha conseguido que los usuarios puedan aportar información a la aplicación sobre las universidades, las ciudades y las asignaturas. Además se ha aportado un mecanismo de interacción entre usuarios por medio de comentarios tanto en la página de cada universidad como en la página de cada asignatura. Sin embargo no se han llegado a crear otros mecanismos como pudieran ser foros o mensajes directos entre usuarios, que permitieran una mayor fluidez en la comunicación entre ellos.
- **Copias de seguridad:** se ha conseguido crear copias de seguridad de la base de datos. Para evitar sobrecargar el servidor, las copias duran como máximo una semana. De esta manera siempre habría 7 copias de la base de datos correspondientes a los últimos 7 días.
- **Migrar el proyecto a un servidor remoto:** se ha alojado la aplicación en un servidor remoto para permitir de una forma sencilla que pudiera ser visitada por otras personas.

6.2. Aplicación de lo aprendido

1. **Fundamentos de la programación:** la primera asignatura de programación de la carrera, donde se aprenden los primeros conceptos básicos de programación. El lenguaje utilizado en esta asignatura fue *Ada*, un lenguaje que no he aplicado en el desarrollo de este proyecto, pero que sirvió para comenzar a aprender a programar.
2. **Sistemas telemáticos y Arquitectura de redes de ordenadores:** han sido útiles porque aprendí el funcionamiento de los protocolos y peticiones HTTP, algo fundamental cuando se desarrolla una aplicación web.

3. **Servicios y aplicaciones telemáticas:** en esta asignatura desarrollé mi primera aplicación web con el *framework Django*. Ha sido de gran ayuda ya que la parte de la gestión del servidor de este proyecto está basada en *Django*.
4. **Ingeniería de sistemas de información:** en esta asignatura comencé a utilizar el lenguaje *Javascript*. Fue una primera toma de contacto con el lenguaje.
5. **Desarrollo de aplicaciones telemáticas:** en esta asignatura terminé de adquirir las capacidades necesarias para poder realizar el lado del cliente con el lenguaje *Javascript*. Con los conocimientos adquiridos en la anterior asignatura, en esta los apliqué y aprendí más sobre la programación web en el lado del cliente. Además aprendí a utilizar distintas *APIs* como la de Google que he utilizado en este proyecto.

6.3. Lecciones aprendidas

He aprendido a desarrollar un proyecto de gran envergadura como lo es éste, y a desarrollar mis capacidades de trabajo y superación, resolviendo los problemas con los que me he ido encontrando. He conseguido fortalecer gran parte de los conocimientos aprendidos en las asignaturas de programación o telemática de la carrera.

Sobre todo he fortalecido los conocimientos de *HTML5*, *CSS3*, *Javascript*, *Django* y peticiones HTTP. Además he aprendido a utilizar más a fondo un *framework* tan útil para el desarrollo web como es *Angular JS*, que empecé a utilizar realizando unas prácticas laborales y me ha permitido desarrollar el *backend* de la aplicación de una manera bastante sencilla.

Y no menos importante, para la redacción de esta memoria ha sido muy útil haber aprendido a redactar con *LaTeX*¹, que a pesar de que puede resultar bastante incómodo, es imprescindible utilizarlo para conseguir una buena presentación formal de la memoria de este TFG.

¹<https://www.latex-project.org/>

6.4. Trabajos futuros

- **Más social:** las aplicaciones y páginas web están en continua evolución hacia un modelo lo más social posible. Se busca la mejor comunicación posible entre los usuarios. Por ello, una línea de trabajo futuro sería la creación de un foro, posibilidad de enviar mensajes directos entre usuarios y la creación de un chat que permita una comunicación más dinámica entre usuarios.
- **Ampliar a más universidades:** esta versión de la aplicación está orientada únicamente para alumnos de una misma universidad. Sería interesante ampliar el *target* de la aplicación y que estuviera orientado a alumnos de todo tipo de universidades y carreras, incluyendo universidades de toda Europa que forman parte del programa Erasmus.
- **Multilinguaje:** debido al multilinguaje europeo, debería traducirse la aplicación a distintos idiomas, como mínimo al inglés.
- **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. El móvil es muy utilizado entre los jóvenes, y una aplicación móvil sería muy útil.
- **Administración:** sería interesante crear un panel de administración más elaborado que el que proporciona *Dango*, que además incluyera la posibilidad de que, debido a que el contenido de la aplicación es generado por los usuarios, un moderador recibiera avisos cuando algún campo de la base de datos haya sido modificado y pudiera revisarlo y controlar que no se escribe nada inoportuno o fuera de lugar.

Bibliografía

- [1] Página sobre erasmus. <http://www.todoerasmus.es/que-es-erasmus/>.
- [2] Página sobre html5. http://www.w3schools.com/html/html5_intro.asp.
- [3] Página sobre css3. http://www.w3schools.com/css/css3_intro.asp/.
- [4] Página sobre bootstrap. <https://raiolanetworks.es/blog/que-es-bootstrap/>.
- [5] Página sobre javascript. <http://www.maestrosdelweb.com/que-es-javascript/>.
- [6] Página sobre angular js. <http://www.desarrolloweb.com/articulos/por-que-angularjs.html>.
- [7] Ari Lerner. *ng-book: The Complete Book on AngularJS*. Fullstack.io, 2013.
- [8] Página sobre git. <http://www.desarrolloweb.com/articulos/introduccion-git-github.html>.
- [9] Página que explica cómo hacer backups. <http://vensign.com/hacer-un-backup-mysql-con-cron-en-linux/>.
- [10] Página con la plantilla utilizada para la interfaz de usuario. <https://graygrids.com/demos/themes/margo/index.html>.
- [11] Página de una librería de angular js para la introducción de textos predictivos. <https://graygrids.com/demos/themes/margo/index.html>.

Apéndice A

Instalación y uso

Para el correcto funcionamiento de la aplicación, lo primero que hay que hacer es ejecutar el archivo *provision.sh*, para que se instalen todos los paquetes que son necesarios, si es que no están ya instalados en la máquina.

Una vez instalados, ejecutar en una *Shell Unix* el comando `python manage.py syncdb`, que creará una base de datos. A continuación hay que ejecutar un comando de la forma

`python manage.py runserver IP:PUERTO`, indicando una dirección IP y un puerto libre: `python manage.py runserver localhost:8000`.

Cuando la aplicación esté en funcionamiento, habrá que introducir en un navegador la *url* `http://localhost:8000`

Apéndice B

Recurso para descargar la aplicación

La aplicación web está disponible en el siguiente repositorio de *Github*:

`https://github.com/marved/tfg`