Práctica de Diseño de Aplicaciones Web

Grupo DAW15

Curso 2014/2015

Índice

Alumnos:	Alejandro Barahona Álvarez Sergio Conde Gómez Ismael Vázquez Fernández
Contacto:	sconde@dilmun.ls.fi.upm.es
Repositorio:	`https://github.com/skgsergio/practica-daw`
Bitácora:	`https://github.com/skgsergio/practica-daw/wiki`

Última modificación:
28 de mayo de 2015

1. Especificaciones de la práctica

Nuestra idea es crear un sistema de gestión y administración de ligas y copas de fútbol. Tendremos información sobre clasificaciones, resultados, sanciones, goles, ... de cada una de las competiciones. También se almacenará información acerca de los distintos clubes que participan en las competiciones.

El sistema distinguirá entre usuarios administradores y usuarios registrados que podrán acceder a competiciones privadas a las que un administrador le invite.

1.1. Definición preliminar de vistas

- **Pantalla de inicio:** Información general donde podrá verse las competiciones públicas (o privadas si el usuario está identificado y está invitado a alguna) en curso o futuras (se planteará la aparición de un calendario con los próximos eventos).
- **Información de competición:** Se mostrarán estadísticas de la competición, clasificación, encuentros, ...
- **Información de club:** Se mostrarán sus estadísticas, próximos encuentros, últimos resultados, ...
- **Información de jugador:** Se mostrarán los datos del jugador (foto, nombre, dorsal, ...), estadísticas de juego (goles, amonestaciones, ...).

En las todas las vistas de información, si el usuario es administrador, se ofrecerá la posibilidad de editar los datos. Además para las competiciones se ofrecerá el acceso a un panel para la introducción de resultados.

1.2. Funcionalidades

La aplicación contara con un sistema de usuarios registrados para la participación directa en el sitio pero a su vez permitirá el acceso anónimo para que los visitantes puedan consultar los datos de las competiciones publicas.

Los administradores son el principal proveedor de contenidos del sistema realizando todas las tareas de introducción de datos, edición de datos, generación de competiciones, etc...

Nuestra idea es afrontar la parte del visionado de datos como una parte sencilla donde la carga se encuentra en la consulta de datos en el servidor y centrar los esfuerzos en la parte de creación y administración de competiciones, equipos y plantillas de forma que sea un proceso rápido, intuitivo y guiado para el usuario con una carga importante en la parte cliente.

1.3. Tecnologías

1.3.1. Parte servidor *propuesta*

- **PHP 5**
- **SQL**

Se valorará el uso de frameworks y librerías como Symfony, Doctrine, ...

El requisito mínimo será un servidor web estándar con soporte para PHP y una instalación de un servidor SQL compatible con MySQL (MariaDB, Percona, ...).

1.3.2. Parte servidor *final*

- **Python 3**[1], con compatibilidad con **Python 2**
- **Django 1.8**[2], framework MVC para python orientado a web
- **MariaDB 10.0**[3]

1.3.3. Parte cliente

- **HTML 5**
- **CSS 3**
- **Bootstrap 3**[4], con el tema *Cosmo*[5]
- **jQuery 1.11**[6]
- **Bootbox 4.3.0**[7], con el estilo (CSS) modificado por nosotros
- **Sorttable 0.5.0**[8], con el estilo (CSS) modificado

por nosotros

- **Font Awesome 4.3.0**[9]

1.3.4. Entornos de programación

Debido a la costumbre de cada uno en su entorno de trabajo usaremos distintos entornos. Por detallar ligeramente vamos a mencionar los principales entornos que cada uno de los miembros usará para desarrollar la práctica:

- **Emacs** con html5-el, web-mode y emmet
- **InteliJ IDEA**
- **Eclipse** con WTP (Web Tools Platform)

1.4. Reparto de tareas *propuesto*

Hemos decidido que todos vamos a intentar colaborar en todas las partes que conformarán la práctica, sin embargo a continuación detallamos los principales responsables de cada parte a modo de reparto:

- **Parte servidor:** Sergio Conde
- **Parte cliente:** Alejandro Barahona, Dan Huang, Ismael Vázquez y Yixuan Wu
  - **Maquetación**
  - **Funcionalidad**

Las siguientes partes se desarrollarán en común:

- **Diseño de la base de datos**

[1] `https://www.python.org/`
[2] `https://www.djangoproject.com/`
[3] `https://mariadb.org/`
[4] `http://getbootstrap.com/`
[5] `https://bootswatch.com/cosmo/`
[6] `https://jquery.com/`
[7] `http://bootboxjs.com/`
[8] `http://github.hubspot.com/sortable/`
[9] `http://fortawesome.github.io/Font-Awesome/`

- **Diseño de la interfaz**
- **Diseño de pruebas**
- **Documentación**

1.5. Reparto de tareas *final*

Aunque se ha tratado colaborar entre todos en todas las partes los encargados principales de cada parte se dividen de la siguiente forma:

- **Parte servidor:**
  - **Diseño de la BBDD:** Alejandro Barahona, Sergio Conde e Ismael Vázquez.
  - **Programación (Python 2/3, Django 1.8):** Sergio Conde.
- **Parte cliente:**
  - **Diseño de la interfaz**: Alejandro Barahona e Ismael Vázquez.
  - **Programación de vistas (HTML, CSS):** Alejandro Barahona.
  - **Funcionalidad (JavaScript)** Alejandro Barahona.
- **Testing de frontend y backend:** Ismael Vázquez.
- **Oras tareas:**
  - **Memoria:** Alejandro Barahona, Sergio Conde e Ismael Vázquez.
  - **Manual de instalación:** Sergio Conde.
  - **Manual de usuario:** Ismael Vázquez.

2. Problemas durante el desarrollo de la práctica

2.1. Cambio de tecnología para la parte del servidor

Llegado el momento de desarrollar la parte servidora nos hemos encontrado con dificultades en el aprendizaje del uso de Symfony. Ante tal problema y viendo que no íbamos a llegar a tiempo de tener la practica finalizada hemos solicitado al tutor el cambio de esta tecnología por Django y Python ya que nos ha sido más fácil su aprendizaje.

Finalmente las tecnologías usadas en la parte del servidor son Python, haciendo uso del framework Django en su versión 1.8.

2.2. Problemas internos del grupo

Llegado el momento de colaborar todos en la práctica dos compañeros, *Dan Huang* y *Yixuan Wu*, no dieron señales de vida. El resto ante tal falta de interés y el retaso causado en la practica debido a ser dos menos y lo expuesto en el punto anterior hablamos con el tutor acerca de este problema el día 12 de Mayo.

El tutor se encargó de enviar un correo a ambos para que diesen su versión de los hechos y hasta donde tenemos conocimiento nunca se pusieron en contacto con el. Sin embargo si se pusieron en contacto con Sergio Conde a través de WhatsApp ese mismo día tratando de resolver el problema. Se les informó del estado de la practica, de los problemas que habíamos tenido con Symfony y del cambio a Django y se les asigno las tareas de testing de frontend y backend.

Se puso una versión de la web con todo el mostrado de datos, tablas, paginación, etc... y tras 4 días sin noticias se quito la versión. El mismo día que la quitamos contactaron con nosotros diciendo que no había fallos y que ya no podían entrar. Se les respondió indicándoles que un miembro del grupo había encontrado un fallo que cuando se pulsaba en cualquier tabla para ver la siguiente página aparecía un error a pantalla completa de Python (por un problema de versiones ya que desarrollábamos en Python 3 y el servidor estaba con Python 2). Tras esto y tras ese mismo día el tutor hablar con nosotros para decirnos que aún no habían hablado con el decidimos retirarles de la práctica.

3. Calendario de desarrollo

- Diseño de bajo nivel de interfaces *<2015-03-16 lun>–<2015-03-26 jue>*
- Diseño de la base de datos *<2015-03-23 lun>*
- Paso a tablas de la base de datos (descartado) *<2015-03-23 lun>*
- Codificación de las interfaces en HTML y CSS *<2015-03-26 jue>–<2015-05-14 jue>*
- Paso a objetos de la base de datos (ORM) *<2015-04-01 mié>*
- Programación del backend en Symfony/PHP (Abandonado) *<2015-04-01 mié>–<2015-05-08 vie>*
- Pruebas de Django/Python *<2015-05-04 lun>–<2015-05-08 vie>*
- Reunión con el tutor para cambiar PHP/Symfony por Django/Python *<2015-05-12 mar>*
- Programación del backend en Django/Python *<2015-05-12 mar>–<2015-05-28 jue>*
- Programación del Javascript y retoques en el HTML para adaptarlo *<2015-05-15 vie>–<2015-05-27 mié>*
- Pruebas sobre el fornted y el backend *<2015-05-14 jue>–<2015-05-28 jue>*

4. Manual de instalación

4.1. Requisitos

Para instalar nuestra aplicación se necesita tener instalado *Python*, dando igual si es versión *2.x* o *3.x*, y *Django* versión *1.8.x*.

Si queremos usar *MySQL* o *MariaDB* necesitaremos instalar también *MySQL-python*[10] si usamos *Python 2.x* o *mysqlclient*[11] si usamos *Python 3.x*.

4.2. Configuración

Una vez se cumplen los requisitos anteriores en la carpeta proyecto editamos el fichero `proyecto/settings.py`. La parte que nos interesa ajustar es `DATABASES`.

Si queremos usar la aplicación web con *MySQL* o *MariaDB* debemos ajustar la variable así:

```
1 DATABASES = {
2     'default': {
3         'ENGINE': 'django.db.backends.mysql',
4         'NAME': 'NOMBRE DE LA BASE DE DATOS',
5         'USER': 'USUARIO DE LA BASE DE DATOS',
6         'PASSWORD': 'PASSWORD DE LA BASE DE DATOS',
```

[10] `https://pypi.python.org/pypi/MySQL-python`
[11] `https://pypi.python.org/pypi/mysqlclient`

```
        'HOST': 'localhost O HOST/IP DE LA BASE DE DATOS',
        'PORT': '3306',
    },
}
```

Otra opción sería usar *PostgreSQL*[12] o incluso *Oracle*[13], que aunque no lo hemos probado debería funcionar sin problemas la aplicación. Para ello debemos ajustar la sección `ENGINE` de la configuración anterior a uno de los siguientes valores teniendo *psycopg2*[14] o *cx_ Oracle* [15] instalado respectivamente:

- `django.db.backends.postgresql_psycopg2`
- `django.db.backends.oracle`

Por último, si preferimos simplificarlo, podemos usar *sqlite3*, aunque no es recomendable ya que no hemos probado la aplicación con el y podría fallar debido a sus limitaciones. Si aún así deseamos intentarlo bastaría con ajustar la variable de la siguiente forma:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

4.3. Inicialización de la aplicación web

Una vez ajustados los parámetros de la base de datos debemos ir a una terminal, situarnos en la carpeta *proyecto*, donde se encuentra el fichero `manage.py` y ejecutar `python manage.py migrate`. Esto creará la base de datos.

Tras crear la base de datos necesitamos crear el primer usuario administrador ejecutando `python manage.py createsuperuser`.

4.4. Servir la aplicación web

Ya tendríamos la aplicación lista para usar, ahora tenemos dos modos de ejecutarla:

- El primero es el más sencillo, que es con el servidor de desarrollo de *Django*. Esto es suficiente para probar la aplicación pero no serviría para el uso público y continuado.
- El segundo método, y el apropiado para publicar la aplicación, sería utilizar un servidor web tal como *Apache*[16] o *Nginx*[17]. Para ello debemos seguir las siguiente guías:
  - *Guía para Apache*: `https://docs.djangoproject.com/en/1.8/howto/deployment/wsgi/modwsgi/`
  - *Guía para Nginx*: `https://docs.djangoproject.com/en/1.8/howto/deployment/wsgi/uwsgi/` y `https://uwsgi.readthedocs.org/en/latest/tutorials/Django_and_nginx.html`

[12] `http://www.postgresql.org/`
[13] `https://www.oracle.com/database/`
[14] `http://initd.org/psycopg/`
[15] `http://cx-oracle.sourceforge.net/`
[16] `http://httpd.apache.org/`
[17] `http://nginx.org/`