VERSIÓN PRELIMINAR DEL PROYECTO

Grupo 48

Fernández Fernández, Iria
Martínez Gómez, Jacobo
Rodríguez Álvarez, Fernando
Vila Fernández, David

Índice

Manual de Instalación

Para realizar el proyecto hemos decidido utilizar el framework CakePHP siguiendo la arquitectura MVC. Para poder probar nuestra aplicación correctamente es necesario instalar un gestor de base de datos, y opcionalmente un servidor Web.

LINUX

Esta instalación se ha probado en una MV Ubuntu 18.04.

Dependencias de PHP necesarias

```
sudo apt-get install php7.2-intl
sudo apt-get install php7.2-mbstring
sudo apt-get install php7.2-mysql
sudo phpenmod pdo_mysql
```

CakePHP

Composer

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') ===
'a5c698ffe4b8e849a443b120cd5ba38043260d5c4023dbf93e1558871f1f07f58274fc6f4c
93bcfd858c6bd0775cd8d1') { echo 'Installer verified'; } else { echo
'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
php composer-setup.php
php -r "unlink('composer-setup.php');"
```

Mover el composer

```
mv composer.phar /usr/local/bin/composer
```

Base de datos

Aunque la configuración de la base de datos se explicara más adelante, es necesario puntualizar que en caso de usar el gestor MySQL hay que tener en cuenta lo siguiente:

En MySQL, a partir de la versión 5.7 no se puede usar el usuario por defecto "root" sin usar la directriz "sudo", lo cual hace imposible a CakePHP conectarse a la BD con ese usuario. Para solucionar esto basta con crear un nuevo usuario y darles permiso1. Un ejemplo de esto sería el siguiente:

```
CREATE USER 'newuser'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON *.* TO 'newuser'@'localhost';
FLUSH PRIVILEGES;
```

[1] Es necesario acordarse tanto del nombre como de la contraseña, ya que serán necesarios para configurar la conexión a la BD. Se explica en la siguiente sección.

Servidor Web

En nuestro caso hemos realizado las pruebas sobre un servidor Apache2. Desaconsejamos usar esta opción ya que existen múltiples problemas con los permisos de acceso a los diferentes documentos y directorios.

La solución a esto es usar el propio servidor de CakePHP, que ya no solo evita estos problemas, si no será capaz de informarnos de las dependencias que puedan llegar a faltar. Para esto basta con lanzar el siguiente comando desde la carpeta del proyecto:

```
bin/cake server
```

De manera predeterminada, el servidor de CakePHP se lanza sobre: http://localhost:8765/

WINDOWS

Esta instalación se ha probado sobre Windows 10 64bits, usando la base de datos MySQL y el servidor Apache integrados en la aplicación XAMPP (También incluye la instalación de PHP).

Dependencias de PHP necesarias

Es necesaria activar la extensión "intl PHP". Para activar esta opción en XAMPP basta con descomentar la línea "`extension=php_intl.dll`" en el archivo "php.ini". En caso de no usar XAMPP, ver la documentación.

CakePHP

En Windows basta con instalar la última versión del "composer".

Servidor Web

Una vez echo esto, podemos mover la carpeta del proyecto en la carpeta del servidor web o podemos lanzar el servidor propio de CakePHP. Una vez mas recomendamos usar el propio servidor de CakePHP para evitar problemas de configuración en el servidor web. Para lanzar este servidor basta con lanzar el siguiente comando desde la carpeta del proyecto:

```
bin/cake server
```

De manera predeterminada, el servidor de CakePHP se lanza sobre: http://localhost:8765/

Estructura del Proyecto

Como explicamos antes usamos CakePHP para implementar nuestra aplicación. CakePHP crea su propia su propia jerarquía de directorios, donde los únicos ficheros y directorios que necesitamos modificar son:

- *"config\app.php"* donde introducimos configuraciones generales, como la conexión a la Base de datos.
- "*src*\" donde implementamos toda la lógica de la aplicación.

El resto de directorios y ficheros del proyecto son generados y usados exclusivamente por CakePHP independiente de la aplicación que se esté desarrollando.

Configuración de la Base de datos

Como ya comentamos antes, la configuración de la base de datos se realiza en el fichero "config\app.php". Dentro de este fichero es necesario ir a la línea 254, donde nos encontramos este array de configuración:

```
'Datasources' => [
    'default' => [
        'className' => Connection::class,
        'driver' => Mysql::class,
        'persistent' => false,
        'host' => 'localhost',
        /*
         * CakePHP will use the default DB port based on the driver selected
         * MySQL on MAMP uses port 8889, MAMP users will want to uncomment
         * the following line and set the port accordingly
         */
        //'port' => 'non_standard_port_number',
        'username' => 'root',
        'password' => '',
        'database' => 'padel48',
        /*
         * You do not need to set this flag to use full utf-8 encoding (internal default since CakePHP 3.6).
         */
        //'encoding' => 'utf8mb4',
        'timezone' => 'UTC',
        'flags' => [],
        'cacheMetadata' => true,
        'log' => false,
```

En nuestro caso al usamos una base de datos MySQL[2] y un servidor Apache, con lo que solo fue necesario modificar los valores de "username", "password" y "database".

[2] CakePHP solo soporta las siguientes BD: MySQL (5.5.3 or greater), MariaDB (5.5 or greater), PostgreSQL, Microsoft SQL Server (2008 or higher)

En caso de usar otra base de datos o necesitar más información sobre estos parámetros de configuración, seria necesario revisar la documentación.

Ahora ya podemos cargar el script SQL (fichero “padel48.sql”), con los datos de la BD y los datos de ejemplo. Antes de cargar el script es necesario desactivar la revisión de claves foráneas para permitir que el script se cargue correctamente.

Lógica de aplicación

Toda la lógica de la aplicación se encuentra en el directorio “src\” siguiendo la arquitectura MVC.

Controladores

Se encuentran en la carpeta “src\Controller”. Todos los controladores son clases PHP.

AppController: Es un controlador que genera CakePHP y nos permite crear funciones globales que son capaces de usar el resto de controladores debido a que todos extienden de esta clase. Contiene funciones para manejar los permisos de los usuarios y para generar arrays con los horarios de las pistas.

CampeonatosController: Es el controlador que gestiona los campeonatos. Permite crear campeonatos y generar los grupos y partidos de este.

CategoriasController: Es el controlador de las categorías. Permite introducir las categorías necesarias para cada campeonato.

EnfrentamientoController: Es el controlador que gestionar los enfrentamientos. Permite visualizar todos los enfrentamientos (solo los referentes a un grupo de un campeonato que estén vinculados con el usuario logueado). También cuenta con la función introducir resultado.

ErrorController: Controlador generado por CakePHP para manejar los errores.

FechasPropuestasController: Es el controlador que gestionar las fechas que proponen los deportistas para jugar los enfrentamientos. Permite desde crear propuestas de fechas, editarlas, borrarlas, aceptarlas y visualizarlas (tanto las propias para editarlas o borrarlas, como las del rival para aceptarlas).

GruposController: Es el controlador encargado de mostrar los grupos. Permite crear y listar todos los grupos de cada campeonato, y el ranking de cada grupo.

HorariosController: Es el controlador encargado de manejar los horarios de las pistas. Permite mostrar los horarios que tienen todas las pistas. También permite modificar estos horarios de manera global (todas las pistas tienen siempre el mismo horario).

NoticasController: Es el controlador encargado de gestionar las noticas. Permite añadir, ver y eliminar noticias.

PagesController: Es un controlador generado por CakePHP para manejar las rutas de los controladores.

ParejasController: Es el controlador encargado de crear las parejas cuando se inscriben a un campeonato.

PartidosController: Es el controlador encargado de gestionar los partidos promocionados. Permite crearlos y borrarlos, así como inscribirse y desinscribirse a ellos.

PistasController: Es el controlador encargado de gestionar las pistas. Permite crear, listas y borrar pistas.

ReservasController: Es el controlador encargado de gestionar las reservas. Permite crear reservas indicando fecha y hora, con un margen de una semana como máximo. El deportista no sabe que pista va a reservar hasta que completa la reserva.

UsuariosController: Es el controlador encargado de gestionar los usuarios. Permite registrarse e iniciar sesión, así como cerrarla. También permite listar todos los usuarios registrados en la aplicación.

Modelos

Los modelos se encuentran en el directorio “src\Model”. En este directorio nos encontramos dos subdirectorios, “Entity” y “Table”. Tantos estos directorios como su contenido con generados por CakePHP al configurar la BD. Y es que CakePHP es capaz de identificar tanto las tablas como las relaciones entre estas de la BD y generar así los modelos.

En el directorio “Entity” nos encontraremos con un fichero por cada tabla de la BD, donde se definen todos sus atributos como propiedades, especificando el tipo de dato que son. También establece si esos atributos serán accesibles o no.

```php
<?php
namespace App\Model\Entity;

use Cake\ORM\Entity;

/**
 * Pareja Entity
 *
 * @property int $id
 * @property string $id_capitan
 * @property string $id_pareja
 * @property int $campeonato_id
 * @property int|null $grupo_id
 * @property int $categoria_id
 * @property int $puntuacion
 * @property bool $clasificado
 *
 * @property \App\Model\Entity\Campeonato $campeonato
 * @property \App\Model\Entity\Grupo $grupo
 * @property \App\Model\Entity\Categoria $categoria
 */
class Pareja extends Entity
{

    protected $_accessible = [
        'id' => true,
        'id_capitan' => true,
        'id_pareja' => true,
        'campeonato_id' => true,
        'grupo_id' => true,
        'categoria_id' => true,
        'puntuacion' => true,
        'clasificado' => true
    ];
```

En el directorio “Table” nos encontramos con un fichero por cada tabla como en “Entity”. Cada fichero representa la estructura de cada tabla de la BD, indicando las relaciones con otras tablas, así como el tipo de relación (1:1, 1:n o n:m).

```php
$this->setTable( table: 'parejas');
$this->setDisplayField( key: 'id');
$this->setPrimaryKey( key: 'id');

$this->belongsTo( associated: 'Campeonatos', [
    'foreignKey' => 'campeonato_id',
    'joinType' => 'INNER'
]);
$this->belongsTo( associated: 'Grupos', [
    'foreignKey' => 'grupo_id'
]);
$this->belongsTo( associated: 'Categorias', [
    'foreignKey' => 'categoria_id',
    'joinType' => 'INNER'
]);
```

También especifica las restricciones de cada columna de la tabla (tipo de dato, longitud, si obligatorio o no, …). Estas restricciones las usa CakePHP en los formularios.

```php
$validator
    ->integer( field: 'id')
    ->allowEmptyString( field: 'id',  message: null,  when: 'create');

$validator
    ->scalar( field: 'id_capitan')
    ->maxLength( field: 'id_capitan',  max: 9)
    ->requirePresence( field: 'id_capitan',  mode: 'create')
    ->notEmptyString( field: 'id_capitan');

$validator
    ->scalar( field: 'id_pareja')
    ->maxLength( field: 'id_pareja',  max: 9)
    ->requirePresence( field: 'id_pareja',  mode: 'create')
    ->notEmptyString( field: 'id_pareja');
```

Por último, también crea restricciones de creación, para traer las restricciones de la BD a la lógica de la aplicación.

```php
public function buildRules(RulesChecker $rules)
{
    $rules->add($rules->existsIn(['id_campeonato'],  table: 'Campeonatos'));
    $rules->add($rules->existsIn(['id_categoria'],  table: 'Categorias'));

    return $rules;
}
```

Dentro de estos fichero se pueden crear funciones personalizadas, como “editarHorarios()” en *HorariosTable* o “addDeportista()” en *PartidosTable.*

Vistas

Las vistas en CakePHP no están en “src\View” como se pensaría en un principio, si no que están en el directorio “src\Template”. Dentro de este directorio nos encontramos un directorio por cada controlador, que contendrá las vistas que este maneja. Por ejemplo, el directorio “Campeonato\” tiene las vistas “add.ctp” e “index.ctp”, que refieren al formulario para añadir un campeonato y al listado de campeonatos de la aplicación respectivamente.

Todos los directorios y ficheros de la carpeta “Template\” siguen esa estructura a excepción del directorio Grupos, donde el fichero “view.ctp” no es la vista particular de un grupo, si no el Ranking de ese grupo, y del directorio “Pages\” que guarda el fichero “home.ctp” con la portada de la aplicación.