

M07 Desarrollo Web en Servidor

UF2 MVC

Mohamed Bouirig Ait Bourig
13-1-2024

ÍNDICE

1 – Objetivos:	2
2 - Entrenadores	3
2.1 - Asociar pokemons	4
2.2 - Ver Pokemons asociados	5
2.3 – Modificar Entrenador	6
2.4 – Eliminar Entrenador	7
3 – Nuevo entrenador	8
4 – Pokemons	9
4.1 – Ver entrenador asociado	9
4.2 – Modificar Pokemon.....	10
4.2 – Eliminar Pokemon	11
5 – Fichero de rutas	12

1 – Objectivos:

INICI

Aquest projecte es basarà en el projecte **MVC** vist a classe, el qual modificarem per completar el **CRUD** i ampliar-lo creant una nova relació **1:N** entre dos noves classes/taules.

Entitats/Models (1 Punt):

1. Entitat Principal (Ex. Articles, Posts, Productes):

- Crea una classe que representi l'entitat principal amb les seves propietats (id, títol, descripció, etc.).
- Afegeix una propietat a la classe que contingui un array d'objectes que representin l'entitat secundària (relació 1:N).

2. Entitat Secundària (Ex. Comentaris, Imatges, Característiques):

- Crea una classe que representi l'entitat secundària amb les seves propietats (id, contingut, url, etc.).

Relació 1:N (1 Punt):

- Estableix una relació 1:N entre les dues entitats. La classe de l'entitat principal ha de contenir una propietat que contingui un *Array* d'objectes de l'entitat secundària.

Vistes i Formularis (1 Punt):

- Modifica les vistes per mostrar els elements de l'entitat principal amb els elements relacionats de l'entitat secundària. Utilitza bucles per recórrer els elements relacionats.
- Crea formularis a les vistes per permetre als usuaris interactuar amb les operacions CRUD. Afegeix camps necessaris per a les dues entitats.

Controladors (1 Punt):

- Implementa mètodes als controladors per gestionar les operacions CRUD per a les dues entitats.
- Afegeix lògica als controladors per manipular les dades i mantenir la coherència en la relació 1:N.

Rutes (0,25 Punts):

- Configura rutes per permetre als usuaris accedir a les diferents operacions CRUD per les dues entitats.

Seguretat (0,25 Punts):

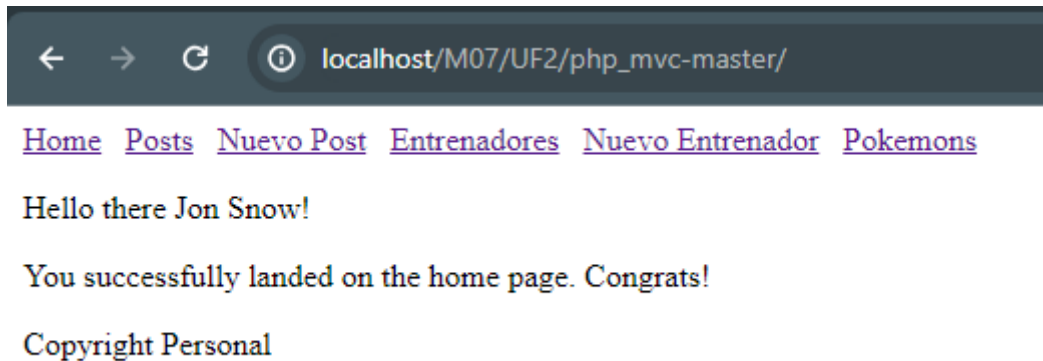
- Implementa mesures de seguretat com validació de dades d'entrada i escapament de dades amb sentències preparades.

Adapta aquests passos al context específic del teu projecte, fent servir els noms de les classes, propietats i mètodes que tinguin sentit per a la teva aplicació.

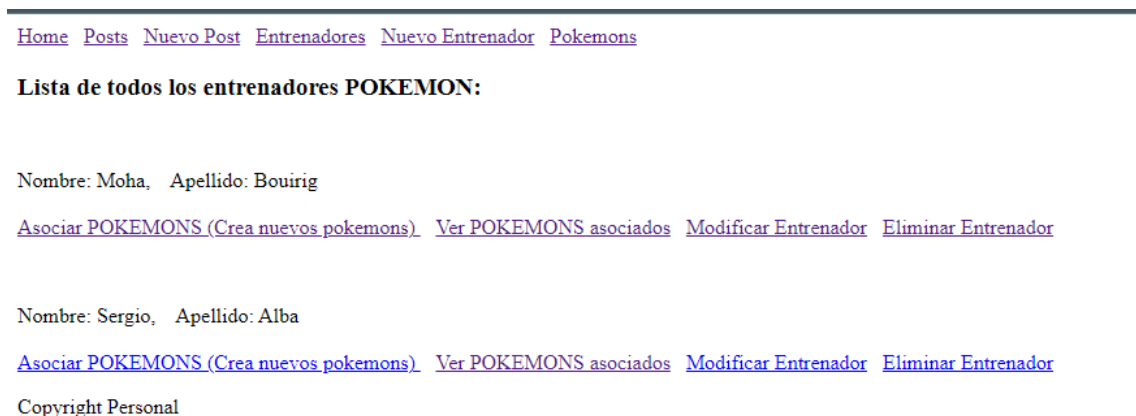
Memòria Obligatoria (0,5 Punts).

2 - Entrenadores

Al acceder a la aplicación web nos aparecerá un menú de navegación:



Si accedemos a Entrenadores podremos crear nuevos pokemons y asociarlos a un entrenador, ver los pokemons asociados, modificar entrenador y eliminar entrenador:



2.1 - Asociar pokemons

En este enlace nos llevara a asociarPokemons.php donde habrá un formulario con los datos a rellenar del pokemon que queramos:

← → ↻ ⓘ localhost/M07/UF2/php_mvc-master/?controller=entrenadores&action=asociarPokemon&id=1

[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Alta nuevo pokemon para el entrenador Moha Bouirig:

Nombre Pokemon:

Tipo Pokemon:

Copyright Personal

```
<h3>Alta nuevo pokemon para el entrenador <?php echo ($entrenador->nombre . " " . $entrenador->apellido); ?></h3>
<form action="?controller=entrenadores&action=guardarPokemons" method="POST">
  <br><br><label for="nombrePokemon">Nombre Pokemon: </label> <input type="text" name="nombrePokemon" id="nombrePokemon">
  <label for="idEntrenador"></label> <input type="hidden" name="idEntrenador" id="idEntrenador" value="<?php echo $entrenador->id ?>">
  <br><br><label for="tipo">Tipo Pokemon: </label> <input type="text" name="tipo" id="tipo">
  <br><br><button type="submit">Guardar</button>
</form>
```

```
// Método estático para mostrar la vista de creación de un nuevo post
public static function asociarPokemon()
{
    if (!isset($_GET['id']))
        return call('pages', 'error');

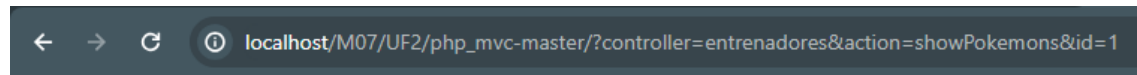
    // Utilizamos el ID proporcionado para obtener el post correspondiente
    $entrenador = Entrenador::find($_GET['id']);
    require_once('views/entrenadores/asociarPokemon.php');
}

public static function guardarPokemons()
{
    if (isset($_POST['nombrePokemon']) && isset($_POST['idEntrenador']) && isset($_POST['tipo'])) {
        $pokemon = Pokemons::guardarNuevoPokemon($_POST['nombrePokemon'], $_POST['idEntrenador'], $_POST['tipo']);
        // Redireccionar después de guardar
        header("Location: ?controller=entrenadores&action=index");
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para guardar un nuevo pokemon
public static function guardarNuevoPokemon($nombrePokemon, $idEntrenador, $tipo)
{
    $db = Db::getInstance();
    $req = $db->prepare('INSERT INTO pokemons (nombrePokemon, idEntrenador, tipo) VALUES (:nombrePokemon, :idEntrenador, :tipo)');
    $req->execute(array('nombrePokemon' => $nombrePokemon, 'idEntrenador' => $idEntrenador, 'tipo' => $tipo));
}
```

2.2 - Ver Pokemons asociados

En este enlace, podremos ver todos los pokemons asociados a este entrenador:



[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Pokemons asociados al entrenador con id 1:

Nombre	Tipo
Pikachu	Electrico
Copyright	Personal

```
// Método para mostrar un post específico
public function showPokemons()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
        return call('pages', 'error');

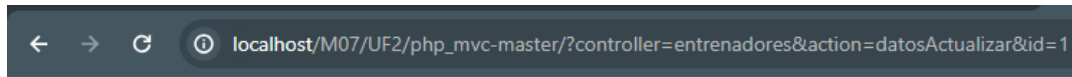
    // Utilizamos el ID proporcionado para obtener el post correspondiente
    $pokemon = Pokemons::find($_GET['id']);
    if ($pokemon != null) {
        require_once('views/pokemons/show.php');
    } else {
        return call('pages', 'error');
    }
}
```

```
<h3>Pokemons asociados al entrenador con id <?php echo $pokemon[0]->idEntrenador; ?></h3>

<table>
    <thead>
        <th>Nombre</th>
        <th>Tipo</th>
    </thead>
    <tbody>
        <?php foreach ($pokemon as $pokemons) { ?>
            <tr>
                <td><?php echo $pokemons->nombrePokemon; ?></td>
                <td><?php echo $pokemons->tipo; ?></td>
            </tr>
        <?php } ?>
    </tbody>
</table>
```

2.3 – Modificar Entrenador

En este enlace podremos modificar todos los datos del entrenador, aparecerá un formulario con los datos actuales del entrenador seleccionado:



[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Modificar Datos Entrenador:

Copyright Personal

```
<p>Modificar Datos Entrenador:</p>

<form action="?controller=entrenadores&action=guardarEntrenador&id=?php echo $entrenador->id ?>" method="POST">
  <br><br><label for="nombre"></label> <input type="text" name="nombre" id="nombre" value="?php echo $entrenador->nombre ?>">
  <br><br><label for="apellido"></label> <input type="text" name="apellido" id="apellido" value="?php echo $entrenador->apellido ?>">
  <br><br><button type="submit">Guardar</button>
</form>
```

```
// Método para mostrar actualizar de un entrenador específico
public function datosActualizar()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
        return call('pages', 'error');

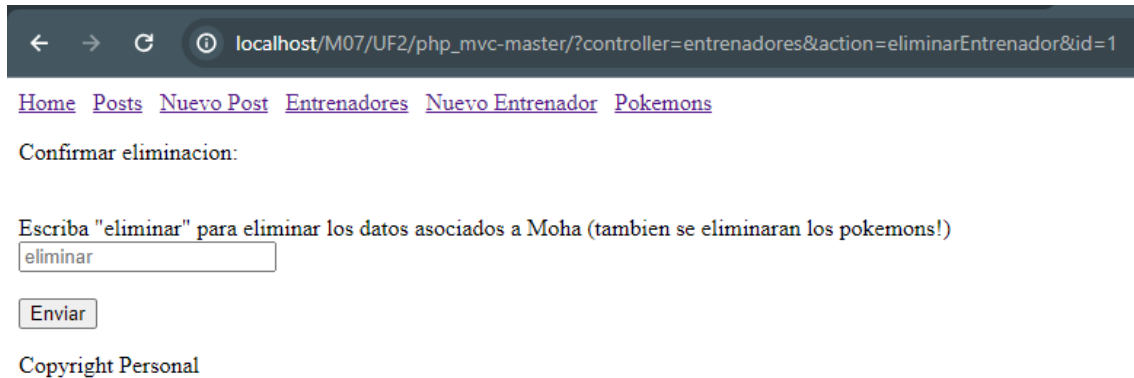
    // Utilizamos el ID proporcionado para obtener el post correspondiente
    $entrenador = Entrenador::find($_GET['id']);
    require_once('views/entrenadores/datosActualizar.php');
}

// Método para actualizar un entrenador específico
public static function guardarEntrenador()
{
    if (isset($_GET['id']) && isset($_POST['nombre']) && isset($_POST['apellido'])) {
        $entrenador = Entrenador::guardarDatosActualizados($_GET['id'], $_POST['nombre'], $_POST['apellido']);
        // Redireccionar después de guardar
        header("Location: ?controller=entrenadores&action=index");
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para actualizar los datos de un entrenador
public static function guardarDatosActualizados($id, $nombre, $apellido)
{
    $db = Db::getInstance();
    $req = $db->prepare('UPDATE entrenadores SET nombre = :nombre, apellido = :apellido WHERE id = :id');
    $req->execute(array('id' => $id, 'nombre' => $nombre, 'apellido' => $apellido));
}
```

2.4 – Eliminar Entrenador

En este enlace podremos eliminar el entrenador seleccionado (También se eliminarán los pokemons asociados):



← → ↻ ⓘ localhost/M07/UF2/php_mvc-master/?controller=entrenadores&action=eliminarEntrenador&id=1

[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Confirmar eliminacion:

Escriba "eliminar" para eliminar los datos asociados a Moha (tambien se eliminaran los pokemons!)

Copyright Personal

```
// Método para mostrar un la eliminación de un entrenador específico
public function eliminarEntrenador()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
        return call('pages', 'error');

    // Utilizamos el ID proporcionado para obtener el post correspondiente
    $entrenador = Entrenador::find($_GET['id']);
    require_once('views/entrenadores/eliminarEntrenador.php');
}

// Método para eliminar un entrenador específico
public static function eliminarOk()
{
    if (isset($_GET['id']) && isset($_POST['eliminar'])) {
        if (strcmp($_POST['eliminar'], 'eliminar') == 0) {
            $entrenador = Entrenador::eliminarConfirmado($_GET['id']);
            // Redireccionar después de guardar
            header("Location: ?controller=entrenadores&action=index");
        } else {
            return call('pages', 'error');
        }
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para eliminar un entrenador
public static function eliminarConfirmado($id)
{
    $db = Db::getInstance();
    $req = $db->prepare('DELETE FROM entrenadores WHERE id = :id');
    $req->execute(array('id' => $id));
}
```


3 – Nuevo entrenador

En este enlace podremos dar de alta a nuevos entrenadores:

← → ↻ ⓘ localhost/M07/UF2/php_mvc-master/?controller=entrenadores&action=crearEntrenador

[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Alta nuevo entrenador:

Copyright Personal

```
<h3>Alta nuevo entrenador:</h3>
<form action="?controller=entrenadores&action=guardar" method="POST">
  <br><br><label for="nombre"></label><input type="text" name="nombre" id="nombre" placeholder="Nombre">
  <br><br><label for="content"></label><input type="text" name="apellido" id="apellido" placeholder="Apellido">
  <br><br><button type="submit">Guardar</button>
</form>
```

```
// Método estático para mostrar la vista de creación de un nuevo post
public static function crearEntrenador()
{
    require_once('views/entrenadores/crearEntrenador.php');
}

public static function guardar()
{
    if (isset($_POST['nombre']) && isset($_POST['apellido'])) {
        $entrenador = Entrenador::guardarNuevoEntrenador($_POST['nombre'], $_POST['apellido']);
        // Redireccionar después de guardar
        header("Location: ?controller=entrenadores&action=index");
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para guardar un nuevo entrenador
public static function guardarNuevoEntrenador($nombre, $apellido)
{
    $db = Db::getInstance();
    $req = $db->prepare('INSERT INTO entrenadores (nombre, apellido) VALUES (:nombre, :apellido)');
    $req->execute(array('nombre' => $nombre, 'apellido' => $apellido));
}
```

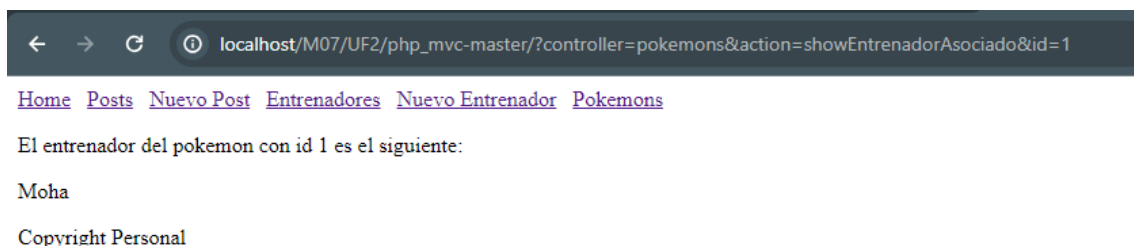
4 – Pokemons

En este enlace nos aparece una lista con todos los Pokemos y podremos ver los entrenadores asociados, modificar los pokemons o eliminarlos:



4.1 – Ver entrenador asociado

En este enlace podremos ver a que entrenador pertenece este pokemon:



```
// Método para mostrar el entrenador asociado a un pokemon específico
public function showEntrenadorAsociado()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
    {
        return call('pages', 'error');
    }

    // Utilizamos el ID proporcionado para obtener el entrenador correspondiente
    $entrenador = Entrenador::find($_GET['id']);
    require_once('views/entrenadores/show.php');
}
```

```
<p>El entrenador del pokemon con id <?php echo $entrenador->id; ?> es el siguiente:</p>

<p><?php echo $entrenador->nombre; ?></p>
```

4.2 – Modificar Pokemon

En este enlace podremos modificar los datos del pokemon menos el id del entrenador ya que esta en hidden:

← → ↻ ⓘ localhost/M07/UF2/php_mvc-master/?controller=pokemons&action=datosActualizar&id=4

[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Modificar Pokemon Pikachu:

Nombre Pokemon:

Tipo Pokemon:

Copyright Personal

```
// Método para mostrar los datos a actualizar de un pokemon específico
public function datosActualizar()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
    {
        return call('pages', 'error');
    }

    // Utilizamos el ID proporcionado para obtener el pokemon correspondiente
    $pokemon = Pokemons::buscarPokemon($_GET['id']);
    require_once('views/pokemons/datosActualizar.php');
}

// Método para guardar la actualización de datos de un pokemon
public static function guardarUpdate()
{
    if (isset($_GET['id']) && isset($_POST['nombrePokemon']) && isset($_POST['idEntrenador']) && isset($_POST['tipo'])) {
        $pokemon = Pokemons::guardarDatosUpdate($_GET['id'], $_POST['nombrePokemon'], $_POST['idEntrenador'], $_POST['tipo']);
        // Redireccionar después de guardar
        header("Location: ?controller=pokemons&action=index");
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para actualizar los datos de un pokemon
public static function guardarDatosUpdate($id, $nombrePokemon, $idEntrenador, $tipo)
{
    $db = Db::getInstance();
    $req = $db->prepare('UPDATE pokemons SET nombrePokemon = :nombrePokemon, idEntrenador = :idEntrenador, tipo = :tipo WHERE id = :id');
    $req->execute(array('id' => $id, 'nombrePokemon' => $nombrePokemon, 'idEntrenador' => $idEntrenador, 'tipo' => $tipo));
}
```

```
datosActualizar.php X
views > pokemons > datosActualizar.php > form
1 <h3>Modificar Pokemon <?php echo $pokemon->nombrePokemon ?></h3>
2
3
4 <form action="?controller=pokemons&action=guardarUpdate&id=?php echo $pokemon->id ?>" method="POST">
5 <br><br><label for="nombrePokemon">Nombre Pokemon: </label> <input type="text" name="nombrePokemon" id="nombrePokemon" value="<?php echo $pokemon->nombrePokemon ?>">
6 <label for="idEntrenador"></label> <input type="hidden" name="idEntrenador" id="idEntrenador" value="<?php echo $pokemon->idEntrenador ?>">
7 <br><br><label for="tipo">Tipo Pokemon: </label> <input type="text" name="tipo" id="tipo" value="<?php echo $pokemon->tipo ?>">
8 <br><br><button type="submit">Guardar</button>
9 </form>
```

4.2 – Eliminar Pokemon

En este enlace podremos eliminar el pokemon seleccionado:

← → ↻ ⓘ localhost/M07/UF2/php_mvc-master/?controller=pokemons&action=eliminarDatos&id=5

[Home](#) [Posts](#) [Nuevo Post](#) [Entrenadores](#) [Nuevo Entrenador](#) [Pokemons](#)

Confirmar eliminacion:

Escriba "eliminar" para eliminar los datos asociados a Raichu

Copyright Personal

```
<p>Confirmar eliminacion:</p>

<form action="?controller=pokemons&action=eliminarOk&id=<?php echo $pokemon->id ?>" method="POST">
  <br><label for="eliminar">Escriba "eliminar" para eliminar los datos asociados a <?php echo $pokemon->nombrePokemon ?></label><br>
  <input type="text" name="eliminar" id="eliminar" placeholder="eliminar">
  <br><br><button type="submit">Enviar</button>
</form>
```

```
// Método para mostrar la confirmación de eliminación de un pokemon
public function eliminarDatos()
{
    // Verificamos si se proporciona un ID en la URL. Sin un ID, redirigimos a la página de error
    if (!isset($_GET['id']))
        return call('pages', 'error');

    // Utilizamos el ID proporcionado para obtener el pokemon correspondiente
    $pokemon = Pokemons::buscarPokemon($_GET['id']);
    require_once('views/pokemons/eliminarDatos.php');
}

// Método para confirmar y eliminar un pokemon
public static function eliminarOk()
{
    if (isset($_GET['id']) && isset($_POST['eliminar'])) {
        if (strcmp($_POST['eliminar'], 'eliminar') == 0) {
            $pokemon = Pokemons::eliminarConfirmado($_GET['id']);
            // Redireccionar después de guardar
            header("Location: ?controller=pokemons&action=index");
        } else {
            return call('pages', 'error');
        }
    } else {
        return call('pages', 'error');
    }
}
```

```
// Método estático para eliminar un pokemon
public static function eliminarConfirmado($id)
{
    $db = Db::getInstance();
    $req = $db->prepare('DELETE FROM pokemons WHERE id = :id');
    $req->execute(array('id' => $id));
}
```

5 – Fichero de rutas

Este es el fichero donde están configuradas todas las rutas:

```
// Inicia el controlador según el tipo de controlador
switch ($controller) {
    case 'pages':
        $controller = new PagesController();
        break;
    case 'posts':
        // Se necesita el modelo para realizar consultas a la base de datos más adelante en el controlador
        require_once('models/post.php');
        $controller = new PostsController();
        break;
    case 'entrenadores':
        // Se necesita el modelo para realizar consultas a la base de datos más adelante en el controlador
        require_once('models/entrenadores.php');
        $controller = new EntrenadoresController();
        break;
    case 'pokemons':
        // Se necesita el modelo para realizar consultas a la base de datos más adelante en el controlador
        require_once('models/pokemons.php');
        $controller = new PokemonsController();
        break;
}

// Ejecuta el método correspondiente en el controlador
$controller->{$action}(); // ejecuta el nombre del método guardado en $action del controlador correspondiente
}

// Definición de controladores y sus acciones
$controllers = array(
    'pages' => ['home', 'error'],
    'posts' => ['index', 'show', 'create', 'guardar', 'datosActualizar', 'guardarUpdate', 'eliminarDatos', 'eliminarOk'],
    'entrenadores' => ['index', 'showPokemons', 'crearEntrenador', 'asociarPokemon', 'guardarPokemons', 'guardar', 'datosActualizar', 'guardarEntrenador', 'eliminarEntrenador', 'eliminarOk'],
    'pokemons' => ['index', 'showEntrenadorAsociado', 'datosActualizar', 'guardarUpdate', 'eliminarDatos', 'eliminarOk']
);
```