ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE
TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

“Diseño e implementación de una plataforma web para la gestión de pistas de pádel.”

AUTOR: Bienvenido Valera Hurtado

DIRECTOR: Francesc Burrul i Mestres

Marzo / 2015

Autor	Bienvenido Valera Hurtado
E-mail del Autor	bienve.valera@gmail.com
Director	Francesc Burrull i Mestres
E-mail del Director	francesc.burrull@upct.es
Título del PFC	Diseño e implementación de una plataforma web para la gestión de pistas de pádel.
Descriptores	Pádel, Reserva de pista de pádel

Resumen

Se plantea el desarrollo de una plataforma web para la reserva de pistas de pádel por un usuario. Para ello se ha diseñado una central de reservas centralizada que permite a los usuarios la búsqueda de pistas libres y una vez elegida su posterior reserva.

Hasta el momento la reserva se realiza en cada club individualmente. Se pretende facilitar al usuario esta tarea de forma que sea más sencillo la práctica de este deporte en auge.

Titulación	Ingeniería Técnica de Telecomunicaciones, Esp. Telemática.
Intensificación	
Departamento	Departamento de Tecnologías de Información y Comunicaciones
Fecha de Presentación	Marzo 2015

Índice

Capítulo 1.

INTRODUCCIÓN.

1.1 Planteamiento inicial y motivación.

Se ha constatado que el pádel es un deporte en auge. Actualmente es el segundo deporte más practicado en España, después del fútbol, desbancando al baloncesto como actividad deportiva. Es por ello que la oferta de pistas para su práctica ha aumentado considerablemente estos últimos años.

El problema surge cuando un usuario desea reservar una pista y para ello tiene que contactar con cada club individualmente, ya sea mediante su rastreo por internet o realizando diversas llamadas telefónicas, hasta localizar un club con pistas disponibles.

Ante la demanda generada se ha pretendido dar solución a la problemática de reservar pista. Para ello se desea implementar un sistema centralizado de reservas vía portal web, que sea utilizado por los jugadores de pádel para la búsqueda y reserva de pistas disponibles según criterios geográficos, de horarios, etc. Tras un rastreo en internet no se ha encontrado ningún servicio de estas características. Hasta ahora las reservas se gestionan individualmente por cada club siendo esto tedioso para el jugador que desea buscar una pista disponible y reservarla.

Las necesidades detectadas actualmente en los usuarios de pádel en internet y los clubs que generan la demanda son las siguientes:

A. Jugadores:

   a) Búsqueda de pistas disponibles entre una amplia oferta de clubes.

   b) Reserva de la pista en el menor tiempo posible.

   c) Toda la información del club localizada en un único portal.

B. Administrador del club:

   a) Un sistema único de reservas

   b) Que los usuarios conozcan su club.

El proyecto consistirá en la creación del sistema de reservas implementado en el portal Servipadel.com.

1.2 Objetivos.

1. Crear una solución que permita a los usuarios encontrar fácilmente los clubs más cercanos.

2. Generar una plataforma de gestión única y centralizada para todos los clubs.

3. Diseñar un buscador sencillo e intuitivo para los usuarios de la aplicación.

4. Construir un sencillo flujo de reserva en pocos pasos.

5. Informar a los jugadores de pádel con movilidad geográfica de los clubs que disponga alrededor, independiente de su ubicación.

6. Facilitar el encuentro entre jugador y club.

7. Formar una comunidad de jugadores de pádel que facilite la práctica de este deporte.

Capítulo 2.

ESTUDIO DE LAS TECNOLOGÍAS Y HERRAMIENTAS DE DESARROLLO.

2.1 Introducción.

Como nuestro objetivo es desarrollar una aplicación web dinámica con acceso a información almacenada en una base de datos, ante la variedad de herramientas disponibles, he elegido el lenguaje de programación **PHP**, la base de datos **MySQL**, y el servidor web **Apache**, por ser estándares "de facto" que ofrecen una potencia y flexibilidad suficientes. He utilizado el servidor **XAMPP** por su sencilla instalación, facilidad de uso e integración de las herramientas indicadas. Además nos proporciona la interfaz para el manejo de la base de datos **phpmyadmin**.

Para el desarrollo del código, he optado por un framework, **Zend Framework 2**, para una mayor velocidad en la elaboración del sitio, mejores garantías para la seguridad del proyecto, y división del código al trabajar con el paradigma de programación MVC (Modelo-Vista-Controlador).

El editor de textos utilizado ha sido el **notepad++**.

2.2 PHP v5.5.9

PHP es un acrónimo recursivo que significa *PHP Hypertext Pre-processor* (inicialmente *PHP Tools*, o, *Personal Home Page Tools*). Fue creado originalmente por Rasmus Lerdorf; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante.

Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el

contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente.

Ventajas:

- Es un lenguaje multiplataforma.

- Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.

- El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.

- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.

- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados extensiones).

Desventaja:

- Como es un lenguaje que se interpreta en ejecución para ciertos usos puede resultar un inconveniente que el código fuente no pueda ser ocultado. La ofuscación es una técnica que puede dificultar la lectura del código pero no la impide y, en ciertos casos, representa un costo en tiempos de ejecución.

Página oficial del proyecto: **php.net**

2.3 MySQL v5.0.11-dev

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

Características:

- El principal objetivo de MySQL es velocidad y robustez.
- Soporta gran cantidad de tipos de datos para las columnas.

- Gran portabilidad entre sistemas, puede trabajar en distintas plataformas y sistemas operativos.
- Cada base de datos cuenta con 3 archivos: uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla.
- Aprovecha la potencia de sistemas multiproceso, gracias a su implementación multihilo.
- Flexible sistema de contraseñas (passwords) y gestión de usuarios, con un muy buen nivel de seguridad en los datos.
- El servidor soporta mensajes de error en distintas lenguas

Ventajas:

- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- Conectividad y seguridad

Desventajas:

- Un gran porcentaje de las utilidades de MySQL no están documentadas.
- No es intuitivo, como otros programas (ACCESS).

Página oficial del proyecto: **http://www.mysql.com/**

2.4 Apache/2.4.7 (Win32)

El **servidor HTTP Apache** es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que Behelendorf quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo

era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años.

Características:

Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.

Apache es una tecnología gratuita de código fuente abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto. Esto le da una transparencia a este software de manera que si queremos ver que es lo que estamos instalando como servidor, lo podemos saber, sin ningún secreto, sin ninguna puerta trasera.

Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que los instalemos cuando los necesitemos. Otra cosa importante es que cualquiera que posea una experiencia decente en la programación de C o Perl puede escribir un modulo para realizar una función determinada.

Apache trabaja con gran cantidad de Perl, PHP y otros lenguajes de script. Perl destaca en el mundo del script y Apache utiliza su parte del pastel de Perl tanto con soporte CGI como con soporte mod perl. También trabaja con Java y páginas jsp. Teniendo todo el soporte que se necesita para tener páginas dinámicas.

Apache te permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

Tiene una alta configurabilidad en la creación y gestión de logs. Apache permite la creación de ficheros de log a medida del administrador, de este modo puedes tener un mayor control sobre lo que sucede en tu servidor.

Página oficial del proyecto: **http://www.apache.org/**

2.5 phpMyAdmin v4.1.6

phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente puede

crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 62 idiomas. Se encuentra disponible bajo la licencia GPL Versión 2.

Características:

- Interface sobre web intuitiva.
- Proporciona herramientas de gestión de la base de datos:
- Edición, creación, modificación y eliminación de bases de datos, tablas, vistas, campos, relaciones e índices.
- Mantenimiento de usuarios y sus privilegios.
- Mantenimiento de procedimientos almacenados.
- Importación de datos desde CSV y SQL.
- Exportación a varios formatos: CSV,SQL, XML, PDF, SO/IEC 26300 - OpenDocument Text y Spreadsheet, Word, LATEX y otros.
- Administración de múltiples servidores.
- Creación del despliegue de la base de datos en un gráfico exportado a PDF.
- Creación de consultas complejas haciendo uso QBE (Query By Example).

Página oficial del proyecto: **http://www.phpmyadmin.net**

2.6 XAMPP v1.8.3

XAMPP es un servidor web multiplataforma, totalmente gratuito y software libre que nos ofrece principalmente un entorno para desarrollar páginas web bajo PHP, MySQL, Perl empleando como servidor Apache.

XAMPP está disponible bajo licencia GNU y es un servidor web libre muy completo, fácil de instalar y configurar con soporte para diferentes sistemas operativos como Windows, Linux o MacOS X.

El paquete dispone de algunas herramientas extra como pueden ser phpMyAdmin para administrar las bases de datos de forma rápida y sencilla. También incluye el módulo OpenSSL por lo que podremos usar **XAMPP en modo HTTPS**.

La instalación es muy sencilla ya sea en Linux o Windows, ya que se puede optar por descomprimir el archivo zip o tar del paquete XAMPP y con dos clicks ya tienes el servidor web funcionando. En Windows también dispone de una versión con instalador. La instalación/desinstalación es una de las características más destacables de este paquete, ya que la sencillez es increíble.

XAMPP está diseñado principalmente para **desarrollar proyectos web o probar proyectos en un servidor local** antes de lanzarlos al público.

Otra característica que hace de XAMPP el entorno ideal de desarrollo web es su panel de control que nos ofrece en todo momento información acerca del estado del servidor y nos permite realizar configuraciones de forma rápida. El panel de control está disponible sólo para la versión de Windows aunque la versión Linux incluye una serie de scripts que también facilitan el manejo y configuración del servidor.

Página oficial del proyecto: **https://www.apachefriends.org**

2.7. Frameworks.

En el siguiente apartado se describirán las ventajas que proporciona desarrollar un proyecto web utilizando un framework. Se realizará una comparativa entre los diferentes frameworks más conocidos.

2.7.1. ¿Qué es un Framework?

En el desarrollo de software, un Framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, librerías y un lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Un "Software Framework" es un diseño reusable de un sistema (o subsistema). Está expresado por un conjunto de clases abstractas y el modo en que sus instancias colaboran para un tipo específico de software. Todos los frameworks de software son diseños orientados a objetos".

2.7.2. Objetivos de un Framework.

Los principales objetivos que persigue un framework son:

- Acelerar el proceso de desarrollo.
- Reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.
- Permitir la utilización de toda la infraestructura existente en cada plataforma (bibliotecas de clases, componentes, etc.).
- Extender el tiempo de vida de una implementación a décadas.
- Desarrollo de software multiplataforma.
- Portabilidad entre plataformas sin pérdida de rendimiento.
- Componentes modulares y abiertos (compilador extensible con "Plugins").

2.7.3. Comparativa Frameworks.

A continuación se estudiarán las ventajas que nos proporcionan los framework más utilizados para el desarrollo web.

2.7.3.1. CodeIgniter.

CodeIgniter es un entorno de desarrollo abierto (licencia Open Source Apache/BSD-style) que permite crear webs dinámicas con PHP. Su principal objetivo es ayudar a que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero. Esto se debe a que dispone de un conjunto bastante amplio de librerías útiles para realizar tareas comúnmente necesarias, así como una interfaz simple y una estructura lógica sencilla para acceder a esas librerías. Entre otras características podemos destacar que es un entorno muy simple. El núcleo del sistema sólo requiere unas pocas librerías para funcionar adecuadamente. Esto supone una gran ventaja frente a otros frameworks de desarrollo que quieren muchos más recursos para realizar las mismas tareas. Las librerías adicionales que se necesiten se cargan de forma dinámica, con lo cual el sistema en sí es muy simple y bastante rápido (está considerado como el framework más rápido, sobretodo en desarrollo bajo PHP).

Ventajas:

- Sistema basado en Modelo-Vista-Controlador.
- Compatible con PHP4.
- Muy liviano.
- Clases de base de datos llenas de características con soporte para varias plataformas.
- Formulario y Validación de datos.
- Seguridad y filtro XSS.
- Manejo de sesión.
- Librería de manipulación de imágenes (cortar, copiar, redimensionar...)
- Larga librería de funciones auxiliares.
- Encriptación de datos.

Desventajas:

- No tiene sistema de plantillas.
- No tiene un layout general.
- No hay módulos.
- Los controladores no cargan por defecto las listas.
- Las vistas no tienen un orden por defecto (ejemplo, controlador/método).
- Hay algunas cosas que no se pueden configurar, y obligan a modificar el núcleo.

2.7.3.2. Symfony

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas Unix como en plataformas Windows.

Ventajas:

- Basado en PHP 5 y, por tanto, lenguaje orientado a objetos.
- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Preparado para aplicaciones empresariales y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Desventajas:

- Primero que nada necesitas por lo menos un VPS para poder publicar tus aplicaciones en la web ya que necesitas tener la habilidad de poder descargare instalar cosas en tu servidor para que symfony funcione apropiadamente.
- El otro problema de Symfony es el caché. Gran parte de la velocidad de Symfony se debe a un uso extensivo del caché por lo que cuando estás desarrollando tiende a ser algo tedioso tener que estar limpiando el caché de vez en cuando.
- Además, los procesos utilizan demasiada memoria.

2.7.3.3. Zend Framework

Zend Framework destaca el hecho de que no sólo busca facilitar la programación a través del patrón MVC, sino también automatizar tareas más específicas, como el acceso a base de datos, el filtrado de datos ingresados a la aplicación o la búsqueda en un sitio web ordenando resultados por relevancia. Zend Framework es un framework de código abierto para desarrollar aplicaciones web y servicios web con PHP 5. Es una implementación que usa código 100% orientado a objetos. La estructura de los componentes es algo único; cada componente está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado. Ofrece un gran rendimiento y una robusta implementación MVC, una abstracción de base de datos fácil de usar, y un componente de formularios que implementa la prestación de formularios HTML, validación y filtrado para que los desarrolladores puedan consolidar todas las operaciones usando de una manera sencilla la interfaz orientada a objetos.

Ventajas:

- Reduce el "time to market" de las aplicaciones, permitiendo ofrecer presupuestos más ajustados.
- Estandariza los procesos más frecuentes, dotándolos de gran robustez.
- Facilita el mantenimiento de las aplicaciones.
- Ofrece muchas facilidades para el acceso a recursos avanzados, que de otro modo resultan bastante más costosos de desarrollar
- A diferencia de otros frameworks, es posible utilizarlo en modo "desacoplado", es decir, aquellas clases o componentes que sean necesarios en cada proyecto, sin arrastrar todo el framework detrás para cualquier pequeña necesidad.
- Tiene el respaldo de la propia ZEND, creadora de PHP, lo que asegura su continuidad futura tanto como la del propio lenguaje PHP.

Desventajas:

- Es necesario comprender algunos patrones de diseño y programación orientada a objetos para utilizar todo el potencial de Zend Framework. Y eso no sólo requiere tiempo dedicado al aprendizaje, sino también la experiencia de trabajar con él.
- Es grande, es pesado, gasta mucha memoria y tiene una gran cantidad de inclusiones.

2.8 Zend Framework 2

Zend Framework es un Framework de código abierto para desarrollar aplicaciones web y servicios web con PHP5. Zend Framework es una implementación que usa código 100% orientado a objetos.

La estructura de los componentes de Zend Framework es algo único. Cada componente está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado.

A menudo se refiere a este tipo de diseño como "use-at-will" (uso a voluntad). Aunque se pueden utilizar de forma individual, los componentes de la biblioteca estándar de Zend Framework conforman un potente y extensible framework de aplicaciones web al combinarse. Zend Framework ofrece un gran rendimiento y una robusta implementación Patrón_Modelo_Vista_Controlador, una abstracción de base de datos fácil de usar, y un componente de formularios que implementa la prestación de formularios HTML, validación y filtrado para que los desarrolladores puedan consolidar todas las operaciones usando de una manera sencilla la interfaz orientada a objetos.

Modelo Vista Controlador

Modelo: ofrece las funcionalidades básicas de la aplicación incluyendo las rutinas de acceso a datos y la lógica de negocios.

Vista: se encarga de generar lo que se presenta al usuario a partir de los datos que recibe del controlador, al mismo tiempo que recogen los datos que brindan los usuarios. Es la parte de la aplicación donde encontrarás el HTML.

Controlador: son los que unen el patrón. Según el pedido del usuario y otras variables ellos pueden decidir ejecutar otro controlador o manipular los datos del modelo para luego asignarle el resultado a una vista en particular. Muchos expertos en MVC recomiendan mantener el controlador lo más limpio posible.

El principal patrocinador del proyecto Zend Framework es Zend Technologies, pero muchas empresas han contribuido con componentes o características importantes para el marco. Empresas como Google, Microsoft y StrikeIron se han asociado con Zend para proporcionar interfaces de servicios web y otras tecnologías que desean poner a disposición de los desarrolladores de Zend Framework.

Página oficial del proyecto: **http://framework.zend.com**

2.9 JavaScript y jQuery.

JavaScript es un lenguaje interpretado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java. Sin embargo, al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia. Es más bien un lenguaje basado en prototipos, porque las nuevas clases se generan clonando las bases (prototipos) y extendiendo su funcionalidad.

Todos los navegadores interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje de una implementación del DOM (Modelo de Objetos del Documento). JavaScript se ejecuta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

jQuery es una biblioteca o framework de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

2.10 CSS.

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación del documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C es el encargado de formular la especificación de las hojas de estilo que servirá de estándar para los agentes de usuario o navegadores.

La idea que se encuentra detrás del desarrollo CSS es separar la estructura de un documento de su presentación. La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML. En este último podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo “style”.

Ventajas:

- Control centralizado de la presentación de un sitio web completo, con lo que se agiliza de forma considerable la actualización del mismo.

- Los navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web remoto, con lo que aumenta considerablemente la accesibilidad.

- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre.

- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

Capítulo 3.

DISEÑO.

3.1 Diagrama de casos de uso.

Los diagramas de casos de uso nos permiten diferenciar los actores que interactúan con nuestra aplicación, las relaciones entre ellos y las acciones que puede realizar cada uno dentro del sistema. Este tipo de diagramas son fácilmente comprensibles tanto por clientes como por usuarios, representan los requisitos funcionales del sistema y se utilizan como base para un desarrollo iterativo e incremental. Los diagramas de casos de uso tienen tres elementos:

- Actores: son los usuarios del sistema. Un actor puede ser una persona, un conjunto de personas, un sistema hardware o un sistema software. Los actores representan un rol, que puede desempeñar alguien que necesita intercambiar información con el sistema.

- Casos de uso: Un caso de uso describe una forma concreta de utilizar parte de la funcionalidad de un sistema. La colección de todos los casos de uso describe toda la funcionalidad del sistema.

- Comunicación entre actores y casos de uso: Cada actor ejecuta un número específico de casos de uso en la aplicación. Por eso decimos que hay comunicación entre actores y casos de uso.

3.1.1 Casos de uso del usuario no registrado.

3.1.1.1 Reserva de pista:

a. Un usuario puede realizar una búsqueda introduciendo en el formulario BÚSQUEDA los parámetros Provincia, Población, Día y Hora que desee. Tras hacer click en el botón BUSCAR PISTA, la vista nos mostrará un listado con los clubs que tienen pistas disponibles e indicará su número.

b. Una vez elegido el club de la lista con pistas disponibles, pulsará el botón RESERVAR PISTA, mostrando en el explorador el horario de cada pista y su disponibilidad. Atendiendo al color marcado se puede consultar la leyenda para diferenciar si está reservada, libre, hora pasada, hora no disponible o es la seleccionada por el usuario.

c. El usuario pulsará CONFIRMAR RESERVA y nos mostrará un formulario que deberá completar el usuario con Nombre y Apellidos, Correo electrónico y Teléfono. A continuación pulsará el botón REALIZAR RESERVA.

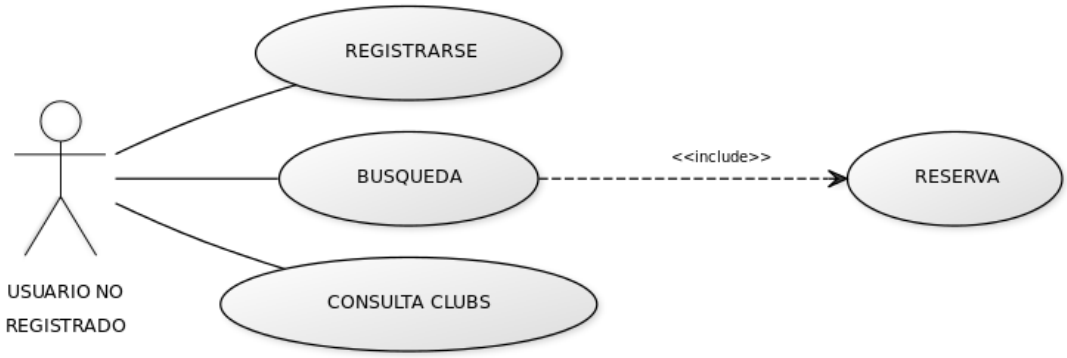
d. La vista que aparecerá a continuación muestra el mensaje “La reserva se ha realizado correctamente” y los enlaces necesarios para Ir a Inicio, Registrarse, Compartir en Facebook y Twitter.

3.1.1.2 Consulta de base de datos de clubs:

a. Se accede a través del menú superior CLUBS, mostrando un listado completo de los clubs actuales.

b. En el formulario situado a la izquierda podrá filtrar los clubs por zona geográfica mostrando el listado conveniente.

c. Podrá seleccionar el club de su interés y se mostrará toda la información del club: Descripción, Fotos, Horario, Localización, página web, teléfono, facebook, etc.

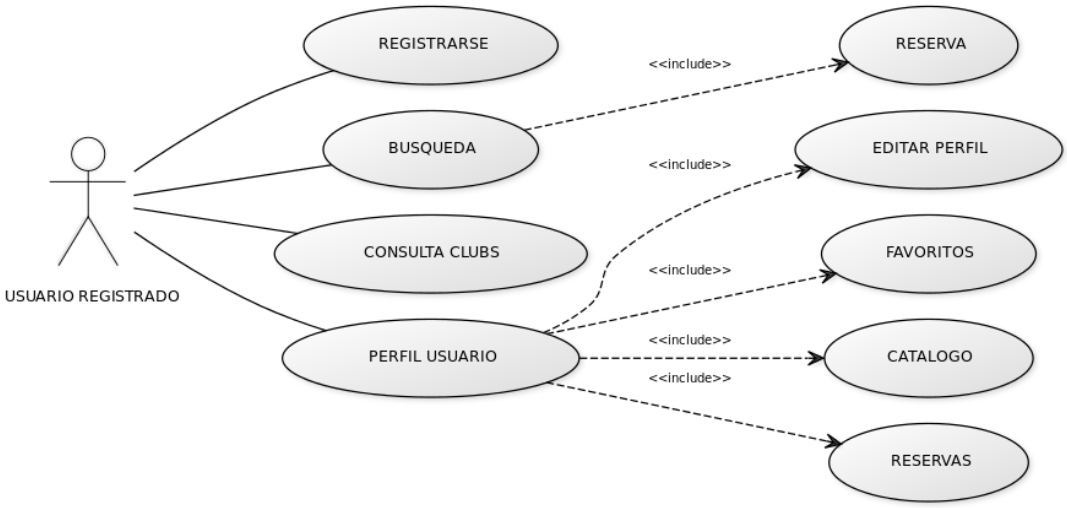


3.1.2 Casos de uso del usuario registrado.

Podrá realizar todas las acciones de un usuario registrado anteriormente descritas con el añadido de poder añadir clubs a favoritos y una vez registrado no necesitará introducir los datos al realizar la reserva.

También podrá consultar el estado de su perfil con las siguientes opciones:

1. El histórico de las reservas realizadas.
2. Consultar datos de sus clubs favoritos.
3. Consultar el catálogo de productos para canjear sus puntos.
4. Editar sus datos personales: imagen, correo electrónico, población provincia, nombre y apellidos.



3.1.3 Casos de uso del gestor del club.

El administrador del club dispondrá de un menú de gestión con las siguientes opciones:

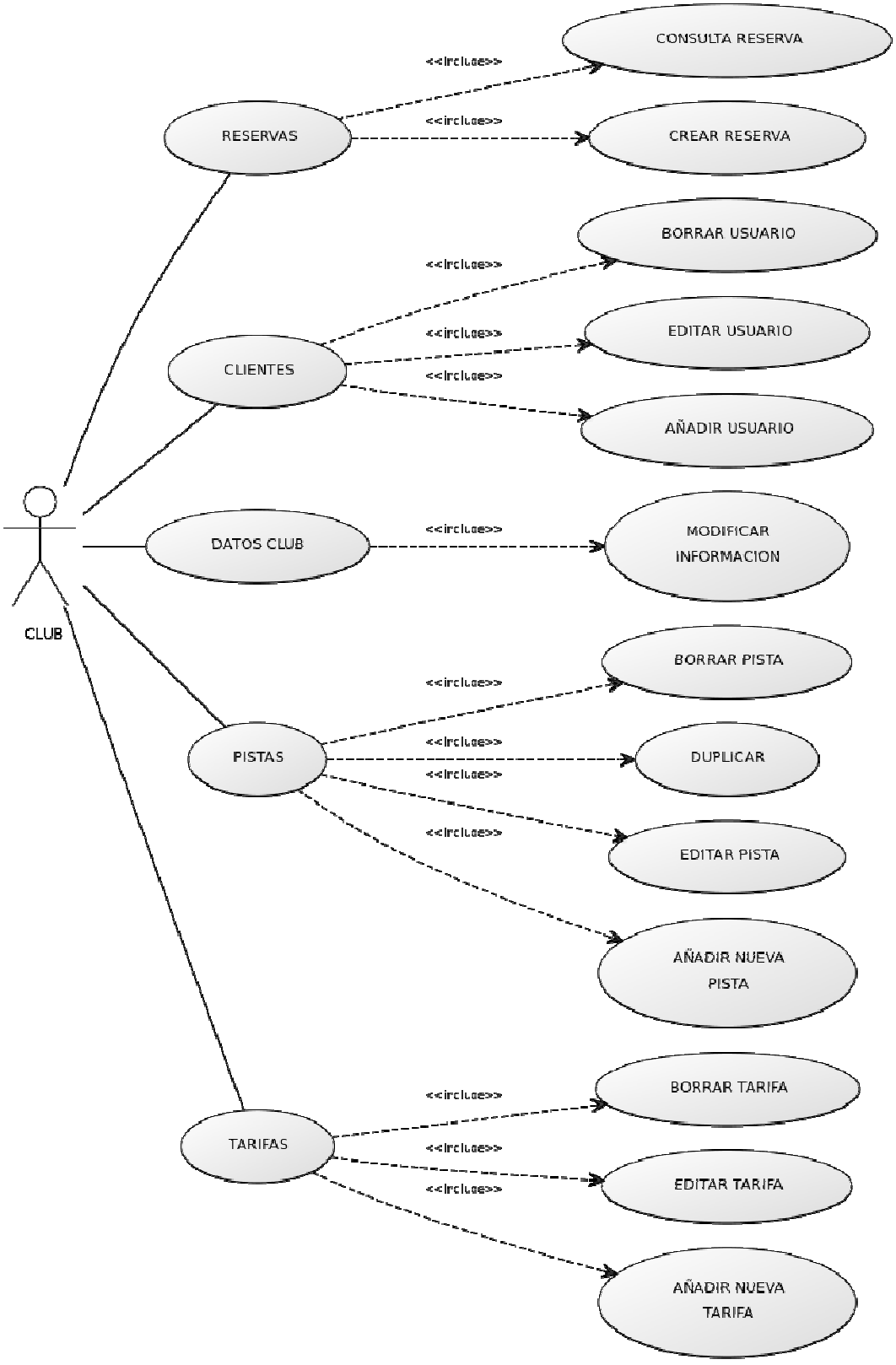
- Reservas: muestra el panel de reservas

- Clientes: muestra los usuarios del club

- Club: muestra los datos del club

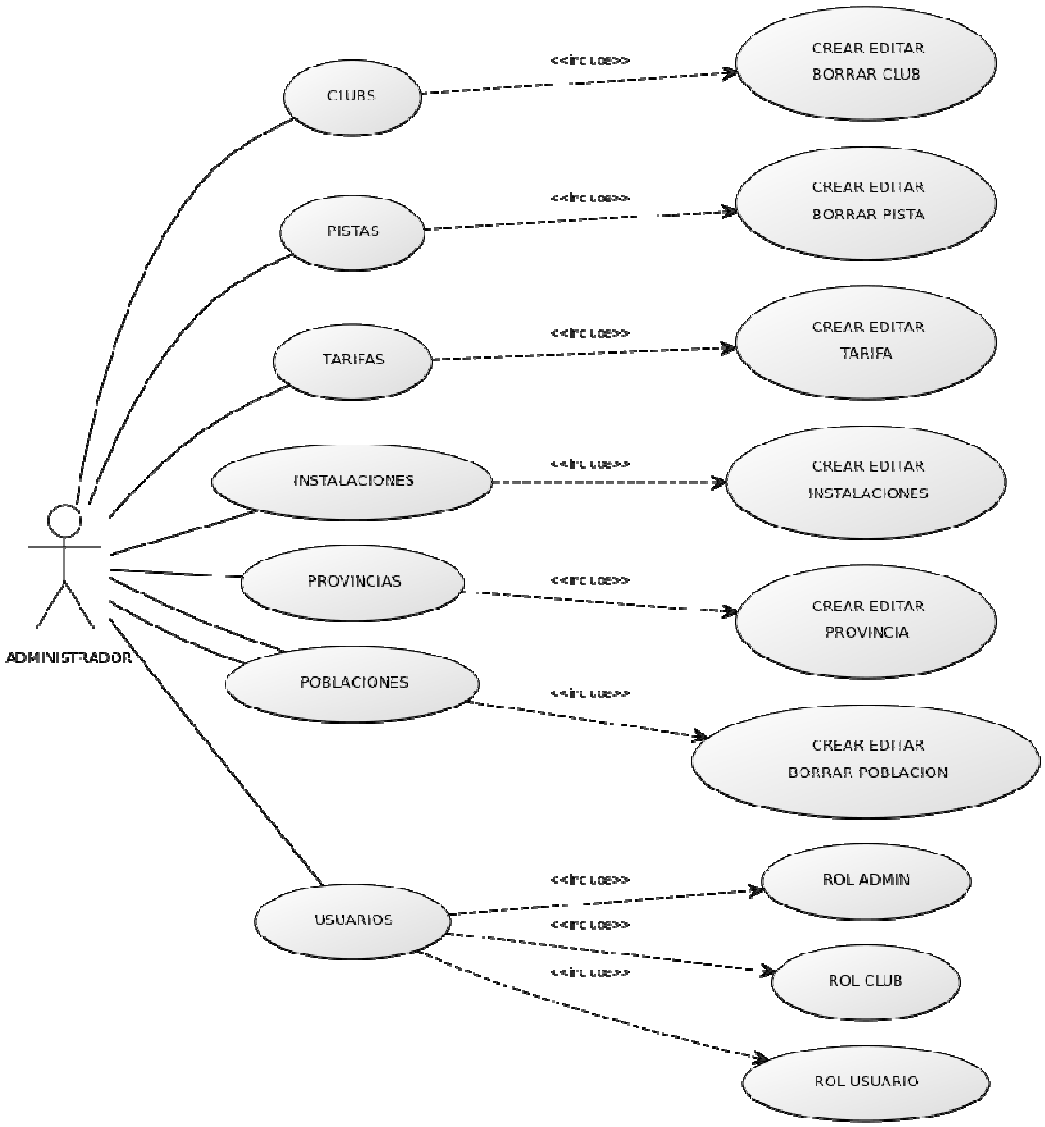
- Pistas: configuración de las pistas

- Tarifas: configuración de las pistas


CONSULTA RESERVA
<<include>>
RESERVAS
<<include>>
CREAR RESERVA
BORRAR USUARIO
<<include>>
CLIENTES
<<include>>
EDITAR USUARIO
<<include>>
AÑADIR USUARIO
CLUB
DATOS CLUB
<<include>>
MODIFICAR INFORMACION
BORRAR PISTA
<<include>>
PISTAS
<<include>>
DUPLICAR
<<include>>
EDITAR PISTA
<<include>>
AÑADIR NUEVA PISTA
BORRAR TARIFA
<<include>>
TARIFAS
<<include>>
EDITAR TARIFA
<<include>>
AÑADIR NUEVA TARIFA

3.1.4 Casos de uso del administrador del portal.

Las funciones más importantes del administrador serán crear los clubs de pádel y asignar el rol a los usuarios según sea usuario registrado, usuario que gestiona un club o un usuario que administre el portal web.

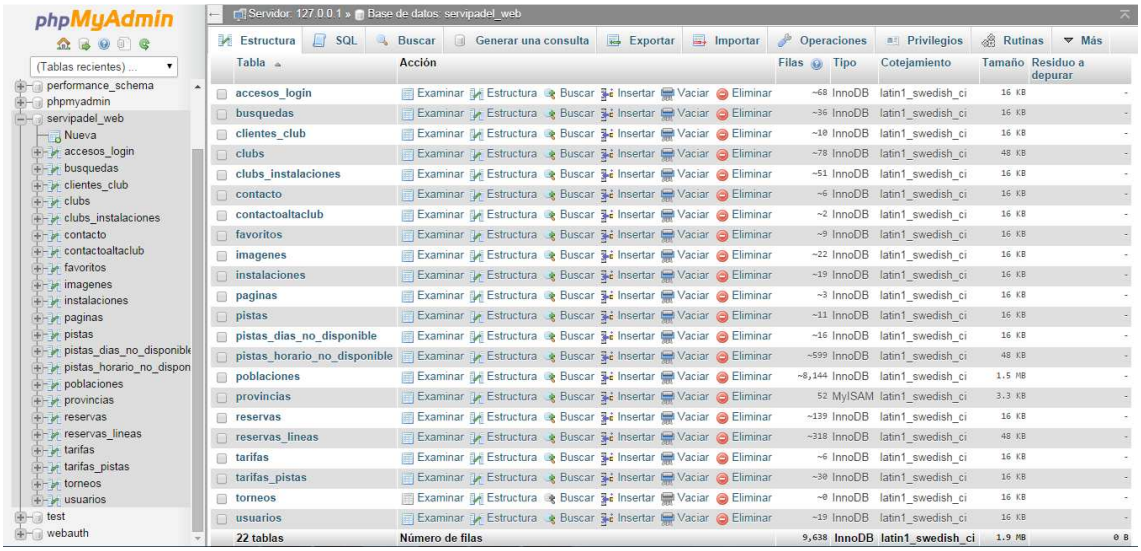


Capítulo 4.

IMPLEMENTACIÓN.

4.1 Base de Datos.

Todos los datos necesarios en la aplicación van a residir en la base de datos servipadel_web creada con PhpMyAdmin.



Para que nuestra base de datos sea accesible se debe configurar. Su configuración se realiza desde el archivo local.php localizado en /servipadel/config/autoload/. Es en este archivo donde se crea una instancia del Zend Db Adapter llamada db. Esta instancia contiene los parámetros de configuración para conectarnos a la base de datos MySQL.

```php
<?php

//Configuración para conectarnos a MySQL con PDO

return array
    (
    'service_manager' => array(
        'factories' => array(
            'Zend\Db\Adapter' => 'Zend\Db\Adapter\AdapterServiceFactory',
        ),
    ),
    'db' => array(
        'username' => 'root',
        'password' => '',
        'driver' => 'Pdo',
        'dsn' => 'mysql:dbname=servipadel_web;host:localhost',
        'driver_options' => array(
            PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES \'utf8\''
        ),
    ),
);
```

username: 'root'→ usuario root de MySQL.

password: ' '→ no hay password configurado

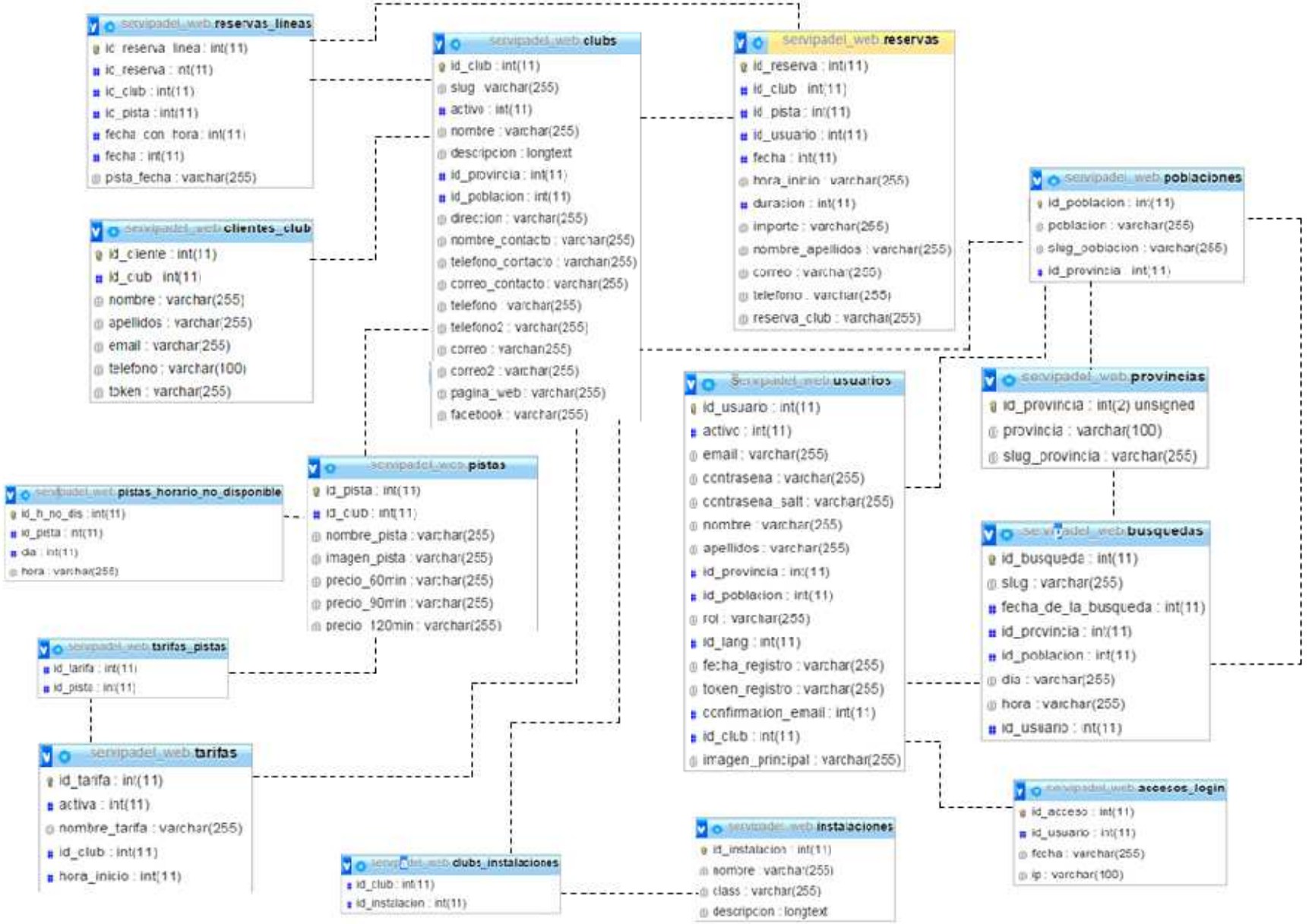
Driver: 'Pdo' →driver necesario para la conexión a cualquier motor de base de datos utilizando una capa de abstracción

dsn: 'mysql:dbname=servipadel_web;host:localhost' →nombre de la base de datos

utf8 → tipo de cotejamiento

Zend Framework ofrece el componente Zend_db, el cual provee de una API para la creación de sentencias SQL. Se pueden crear sentencias SQL de dos formas principalmente. La primera es a través de la creación de un objeto select mediante el método select() del DbAdapter, desde el que luego pueden ir seteando individualmente cada parte de la sentencia (from, where, etc.).

La otra forma es a través del método fetchAll(). Se escribe una consulta SQL que se pasa como parámetro del método y éste devuelve un array con las filas devueltas por la consulta. Si se quiere obtener toda la información de una tabla (todas las filas) basta con ejecutar fetchAll(), sin ningún parámetro. Éste método también ofrece la posibilidad de pasarle como parámetros las siguientes variables: $where, $order, $count, $offset.


servipadel_web.reservas_lineas
id_reserva_linea : int(11)
id_reserva : int(11)
id_club : int(11)
id_pista : int(11)
fecha_con_hora : int(11)
fecha : int(11)
pista_fecha : varchar(255)
servipadel_web.clubs
id_club : int(11)
slug : varchar(255)
activo : int(11)
nombre : varchar(255)
descripcion : longtext
id_provincia : int(11)
id_poblacion : int(11)
direccion : varchar(255)
nombre_contacto : varchar(255)
telefono_contacto : varchar(255)
correo_contacto : varchar(255)
telefono : varchar(255)
telefono2 : varchar(255)
correo : varchar(255)
correo2 : varchar(255)
pagina_web : varchar(255)
facebook : varchar(255)
servipadel_web.reservas
id_reserva : int(11)
id_club : int(11)
id_pista : int(11)
id_usuario : int(11)
fecha : int(11)
hora_inicio : varchar(255)
duracion : int(11)
importe : varchar(255)
nombre_apellidos : varchar(255)
correo : varchar(255)
telefono : varchar(255)
reserva_club : varchar(255)
servipadel_web.poblaciones
id_poblacion : int(11)
poblacion : varchar(255)
slug_poblacion : varchar(255)
id_provincia : int(11)
servipadel_web.clientes_club
id_cliente : int(11)
id_club : int(11)
nombre : varchar(255)
apellidos : varchar(255)
email : varchar(255)
telefono : varchar(100)
token : varchar(255)
servipadel_web.usuarios
id_usuario : int(11)
activo : int(11)
email : varchar(255)
ccontrasena : varchar(255)
ccontrasena_salt : varchar(255)
nombre : varchar(255)
apellidos : varchar(255)
id_provincia : int(11)
id_poblacion : int(11)
rol : varchar(255)
id_lang : int(11)
fecha_registro : varchar(255)
token_registro : varchar(255)
confirmation_email : int(11)
id_club : int(11)
imagen_principal : varchar(255)
servipadel_web.provincias
id_provincia : int(2) unsigned
provincia : varchar(100)
slug_provincia : varchar(255)
servipadel_web.pistas
id_pista : int(11)
id_club : int(11)
nombre_pista : varchar(255)
imagen_pista : varchar(255)
precio_60min : varchar(255)
precio_90min : varchar(255)
precio_120min : varchar(255)
servipadel_web.pistas_horario_no_disponible
id_h_no_dis : int(11)
id_pista : int(11)
dia : int(11)
hora : varchar(255)
servipadel_web.busquedas
id_busqueda : int(11)
slug : varchar(255)
fecha_de_la_busqueda : int(11)
id_provincia : int(11)
id_poblacion : int(11)
dia : varchar(255)
hora : varchar(255)
id_usuario : int(11)
servipadel_web.tarifas_pistas
id_tarifa : int(11)
id_pista : int(11)
servipadel_web.tarifas
id_tarifa : int(11)
activa : int(11)
nombre_tarifa : varchar(255)
id_club : int(11)
hora_inicio : int(11)
servipadel_web.accesos_login
id_acceso : int(11)
id_usuario : int(11)
fecha : varchar(255)
ip : varchar(100)
servipadel_web.clubs_instalaciones
id_club : int(11)
id_instalacion : int(11)
servipadel_web.instalaciones
id_instalacion : int(11)
nombre : varchar(255)
class : varchar(255)
descripcion : longtext

4.2. Arquitectura de Zend Framework 2.

En este proyecto se va a usar el patrón de arquitectura software MVC (Modelo-Vista_Controlador) que nos ofrece Zend Framework 2. Siguiendo este patrón conseguimos una división de la aplicación web en tres partes, una de ellas se encarga de la interfaz de usuario, otra del tratamiento de los datos y la lógica de negocio y por último, la tercera es la encargada de comunicar las otras dos partes

Modelo (*Model*): esta parte es la encargada de los datos, la que se va a comunicar con la base de datos. Existen varias formas de gestionar esta parte, una de ellas es crear un Modelo por cada módulo o pieza de la aplicación web, de esta forma un mismo Modelo controlará varias tablas. Otra forma es crear para cada tabla de la base de datos una clase que será su Modelo, aunque en algún caso pueda a llegar a controlar varias tablas que tengan mucho en común.

Vista (*View*): es la parte que maneja lo que verá el usuario de la aplicación web. Transforma todos los datos recibidos por el controlador, para que tengan un formato y una estructura adecuados para el usuario.

Controlador (*Controller*): parte que comunica las otras dos, ya que éstas deben estar aisladas. Se encarga de recibir las acciones del usuario e invocar peticiones al Modelo. Procesa todos los datos necesarios y los manda a la vista mediante la asignación de variables.

En la carpeta *Application* es donde reside todo el código de la aplicación y es la encargada de manejar el patrón MVC, explicado anteriormente, a través de sus tres carpetas principales model, controller y views:

Controller: en esta carpeta irán alojados los distintos controladores que se creen para la aplicación. Un controlador es una clase que extiende de AbstractActionController y esta clase contiene métodos llamados acciones, estos métodos especiales tienen el nombre seguido del sufijo *Action*. Los controladores son archivos con extensión .php. El controlador por defecto que se ejecuta es *IndexController.php* si no se indica ningún controlador. El

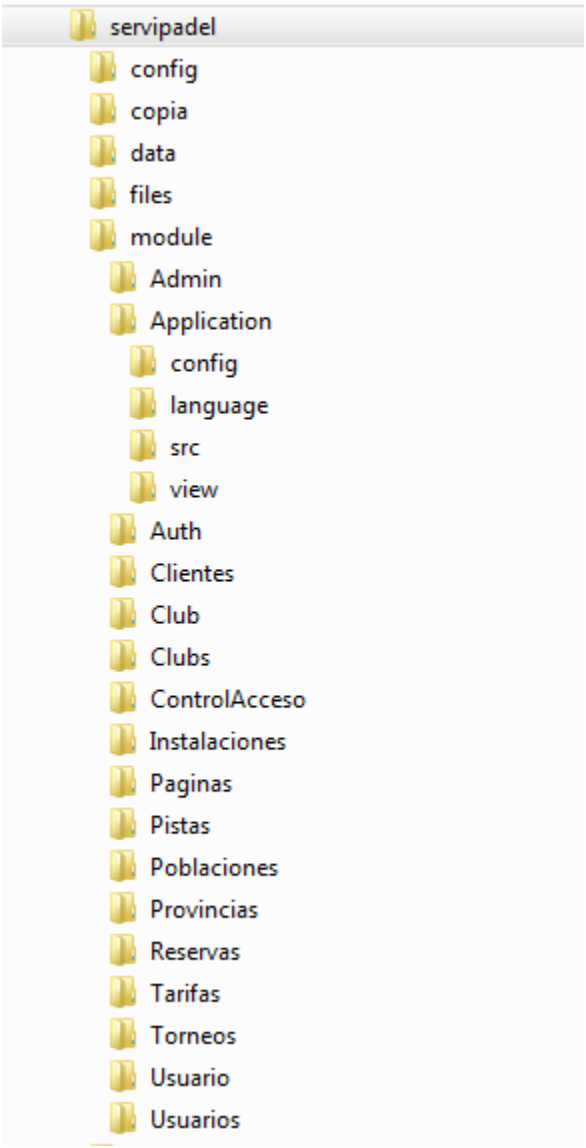
controlador *ErrorController.php* se ejecutará cada vez que queramos acceder a una página que no existe o se produzca algún error en la aplicación. Las URL`s en ZF2 tienen la forma */controlador/acción*, esta URL ejecutará la acción del controlador indicada.

Model: aquí irán todas las clases que creemos como modelos de nuestra aplicación.

Views: carpeta encargada de guardar las vistas, las páginas con HTML. Aquí es donde se genera la fachada de la interfaz de usuario, lo que ve el usuario.

4.2.1 Estructura de directorios.

Al implementar este sistema con Zend Framework 2 se ha utilizado la arquitectura de directorios que usa este framework.



Todo el proyecto cuelga de la carpeta con el nombre del proyecto servipadel. A parte del proyecto también se crean unas librerías de Zend Framework 2.

4.2.2 Componentes del patrón MVC.

Zend Framework sigue el patrón de arquitectura MVC, como ya se ha explicado anteriormente. Se pretende explicar con más detalle cada uno de los tres componentes de Zend Framework, como implementar cada uno y la comunicación entre ellos, es decir, cómo se puede pasar información de unos a otros.

Modelo

El modelo de la aplicación va a residir en servipadel/module/, donde se guardarán todos los modelos (clases) necesarios. Se debe crear un archivo .php por cada tabla a usar de la base de datos, nombrando cada archivo con el mismo nombre que tiene la tabla. Por otro lado se debe crear un archivo por cada controlador establecido en la aplicación.

Cada modelo es una clase que extiende de Zend_Db_Adapter. En cada clase se definen las funciones necesarias. En los modelos relacionados con las tablas de la base de datos lo primero que hay que definir dentro de la clase es una variable protegida con el nombre de la tabla que vamos a extraer información y a continuación las funciones.

```php
<?php

namespace Clientes\Model\Entity;

use Zend\Db\TableGateway\TableGateway;
use Zend\Db\Sql\Select;
use Zend\Db\Sql\Sql;
use Zend\Db\Sql\Predicate\Expression;

class ClienteTable {

    protected $tableGateway;

    public function __construct(TableGateway $tableGateway) {

        $this->tableGateway = $tableGateway;
    }

    public function fetchAll() {
        $resultSet = $this->tableGateway->select();
        return $resultSet;
    }
```

Ejemplo clase del modelo

Es necesario crear la variable protegida para que así el modelo sepa de qué tabla de la Base de Datos tiene que coger o modificar información. Así cada vez que necesitemos referenciar un campo de la tabla basta con escribir el nombre de ese campo. En el anterior ejemplo está definida la función fetchAll() que devuelve las filas seleccionadas por la consulta select.

Controlador

Los controladores de la aplicación residen en Appllication/Controller y sirven para comunicar el modelo con la vista. Al crear el proyecto se crean los controladores ErrorController.php e IndexController.php, siendo el controlador Error donde se detallarán los posibles errores de la aplicación. La aplicación constará de un controlador por cada módulo de

la aplicación, para toda la estructura general se empleará el controlador por defecto IndexController.php, donde estará todo lo relacionado con la cabecera, el menú, etc. La parte de autenticación se encuentra dentro de la estructura general pero se considera lo suficientemente compleja como para crear un módulo aparte llamado Auth.

Vista

Las vistas se encuentran en el directorio Application/view. Son archivos .phtml, ya que ahí se mezcla código php, html y JavaScript o referencias a archivos JavaSript. Las vistas no se crean porque sí, si no que existirá una vista por cada acción de cada controlador. Para cada controlador se crea una carpeta en el directorio de las vistas y dentro de cada una de esas carpetas estarán los archivos .phtml, uno por cada acción. El nombre de la vista es el mismo que el nombre de la acción a la que hacen referencia.

4.3 Capa de presentación.

A continuación se va a inspeccionar el botón creado en la parte superior derecha RESERVA PISTAS ONLINE. Dicho botón está formado por las clases btn, btn-sm, btn-success y btn-reservas. Las clases las clases btn, btn-success y btn-reservas están definidas en el archivo style.css, ubicado en servipadel/public/css, mientras que la clase btn-sm se define en el archivo bootstrap.min.css, localizado en el mismo directorio que style.css.

```
.btn{
    padding: 6px;
}

 .btn-success{
    background: #4ca83f;
    border: #4ca83f;

}

.btn-reservas {
    float: right;
    margin: 20px 0px 20px 35px;
    padding: 5px 20px;
    font-weight: 300;
}
```

Las propiedades de definen estas clases son:

- padding: establece la anchura
- background: estable el color de fondo
- border: establece algunas o todas las propiedades de los bordes de los elementos
- float: posiciona el botón a la derecha
- margin: establece la anchura de alguno o todos los márgenes de los elementos
- font-weight: establece el color de la letra

4.4 Código ejecutado en cliente.

Con JavaScript conseguimos que el código se ejecute en el cliente. La función onclick devuelve el método JavaScript *seccionar()* localizada en un script del archivo *Application/view/application/reservas/index.phtml.*

Al método *seccionar* se le pasan los parámetros *id_pista* y *hora*. Este método obtiene el tiempo de juego seleccionado anteriormente y calcula todos los datos del resumen de la RESERVA situado en el margen derecho.

Capítulo 5.

FUNCIONES IMPLEMENTADAS.

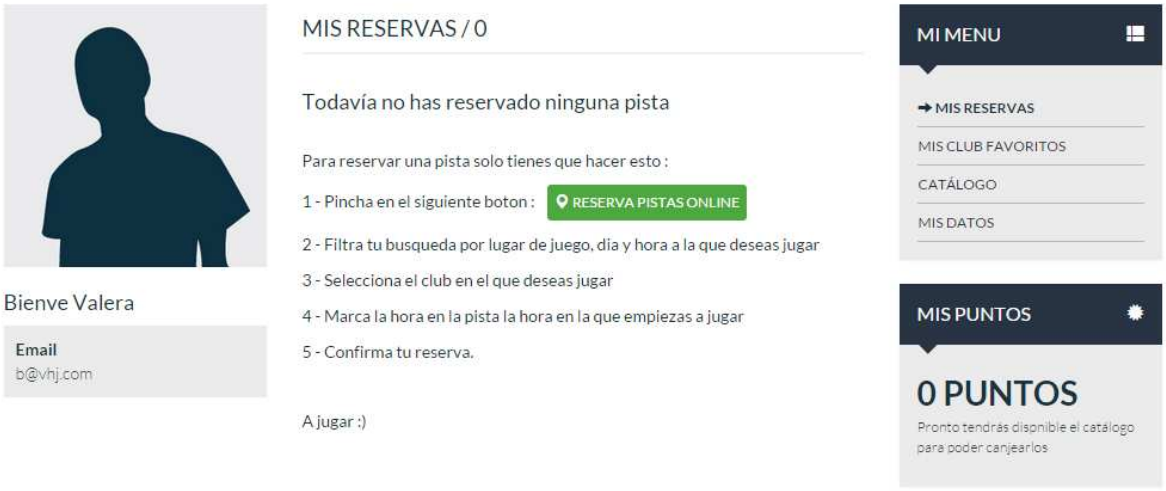
5.1 Registro de un usuario.

El botón REGISTRATE en la parte superior derecha mapea la URL http://localhost/servipadel/usuario/registro ejecutando el método *indexAction()* localizado en el módulo *Auth\Controller\RegistrationController.php*

```
'registro' => array(
    'type' => 'Zend\Mvc\Router\Http\Literal',
    'options' => array(
        'route' => '/usuario/registro',
        'defaults' => array(
            'controller' => 'Auth\Controller\Registration',
            'action' => 'index',
        ),
    ),
),
```

Dicho método devolverá la vista index.phtml del módulo *registration* mostrando el formulario RegistrationForm.php creado en el controlador.

El usuario rellenará el formulario y pulsará el botón REGISTRATE que ejecutará el método *indexAction()* situado en el controlador *IndexController.php* del módulo Usuario. Carga las vistas para mostrar en el navegador el Perfil del usuario.

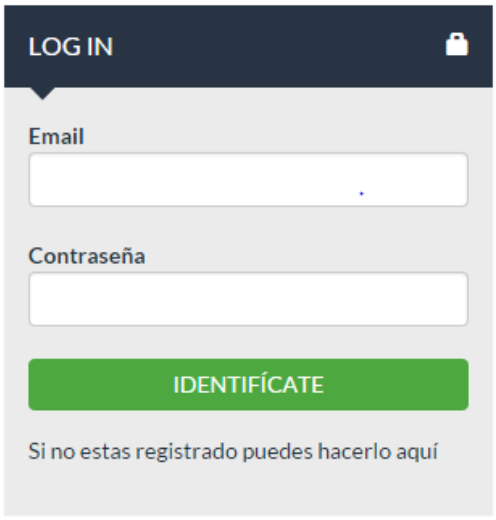


Todas las acciones que puede realizar el usuario desde su panel de administración se encuentran localizadas en este último controlador *IndexController.php* del módulo Usuario. Los métodos que se han implementado son:

a) indexAction()→ Muestra el listado de reservas del usuario.
b) favoritosAction()→ Muestra una lista de los clubs que el usuario ha añadido como club favorito, consultando en la base de datos las tablas favoritos, usuarios y clubs.
c) catalogoAction()→Mostraría el catálogo en cuanto esté disponible, mientras mostrará el mensaje “Estamos trabajando en el catálogo para que puedas canjearlos los puntos por productos de pádel” implementado en la vista catalogo.phtml
d) editarAction()→Si el usuario quisiera editar sus datos, se modificarían los que deseara al pulsar el botón.
e) subirImagenAction()→ Con el que el usuario puede subir una imagen insertando en la tabla usuarios la columna imagen_principal.

5.2 Log in de un usuario

Si el usuario se ha registrado previamente puede acceder a través del botón LOG IN en la parte superior derecha que nos mostrará el formulario de acceso con el método *loginAction()*, ubicado en *IndexController.php* en el modelo *Auth*, que muestra en la vista *login.phtml* el formulario *AuthForm.php.*



El botón IDENTIFÍCATE nos ejecuta el método *loginAction()* que comprobará en la tabla usuarios si existen los datos introducidos. En el caso de no corresponder con ningún usuario se devuelve el mensaje “El email o la contraseña no son correctas”. Si la comprobación se realiza con éxito, examina el rol que tiene el usuario y redirecciona la ruta, según sea usuario, gestor de un club o administrador, al controlador correspondiente.

```
$this->getUsuarioTable()->insertAcceso($authAdapter->getResultRowObject()->id_usuario);
switch ($authAdapter->getResultRowObject()->rol) {
    case 'usuario':
        return $this->redirect()->toRoute('usuario', array('controller' => 'index'));
        break;
    case 'club':
        return $this->redirect()->toRoute('gestionClub', array('controller' => 'index'));
        break;
    case 'admin':
        return $this->redirect()->toRoute('admin', array('controller' => 'index', 'action' => 'index'));
        break;
}

break;
```

5.3 Búsqueda y reserva de una pista.

Se accederá al portal web con la URL http://localhost/servipadel/. Nos mostrará la página inicial del buscador. Según el modelo MVC con el que trabaja Zend Framework 2, al introducir esta URL se dirige al archivo por defecto IndexControlller.php, dado que no se indica otro controlador en la URL, situado en servipadel/module/Application/src/Application/Controller/.

Dentro de este controlador realiza la función:

```
public function indexAction() {
```

Dicha función actualiza los datos fecha y hora a los actuales mostrando esos datos en el formulario de búsqueda para que al usuario le sea más fácil completarlo.

```
$formBuscador = new BuscadorForm();

/// MARCO EL DIA EN EL FORMULARIO
$hoy = date("d/m/Y", time());
$formBuscador->get('dia')->setValue($hoy);

$array_fecha = explode("/", $hoy);
$dia_time = mktime(0, 0, 0, $array_fecha[1], $array_fecha[0], $array_fecha[2]);

$min_fin = 24 * 60;
for ($index = 0; $index < $min_fin; $index = $index + 30) {
    $time = $dia_time + ($index * 60);
    if ($time > time()) {
        $hora_señal = $index;
        break;
    }
}
$formBuscador->get('hora')->setValue($hora_señal);
```

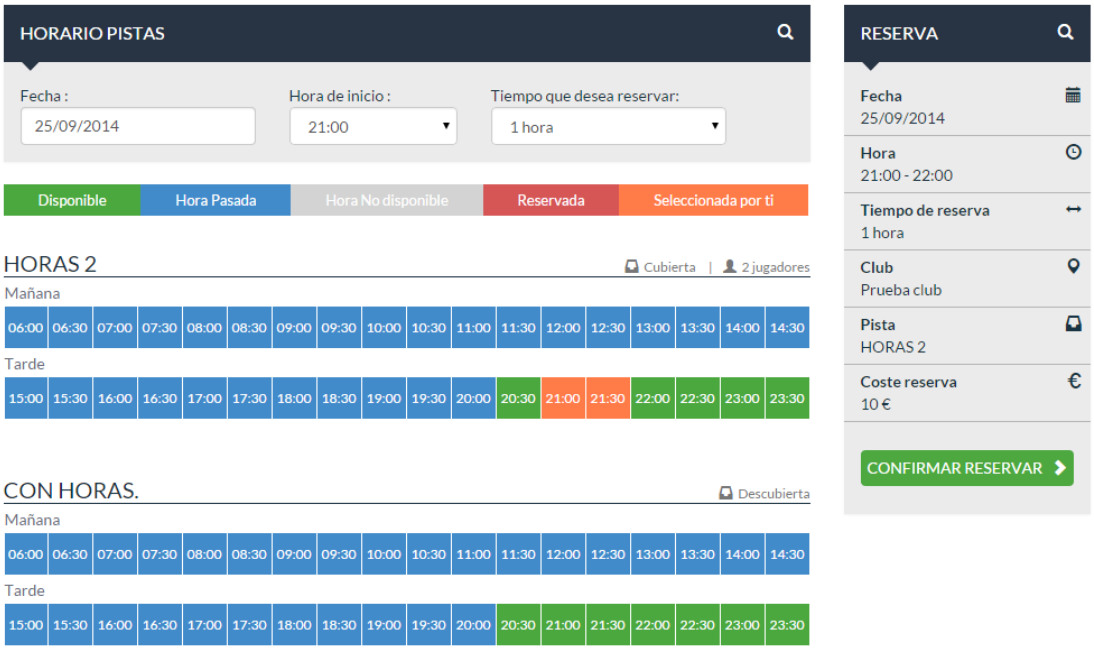
En el desplegable de Provincias se añaden sólo las provincias que tienen club con pistas abiertas. A continuación se actualizan las poblaciones, mostrando únicamente una selección de poblaciones que tienen pistas. Finalmente éste método devuelve toda esta información a la vista que se presentará en el explorador.

Tras hacer click en el botón de BUSCAR PISTA del formulario de BUSQUEDA, devuelve un listado de clubs con el número de pistas disponibles. Esta acción es mapeada por el router como buscador-pistas-de-padel. Esta URL está localizada en el archivo *module.config.php.*

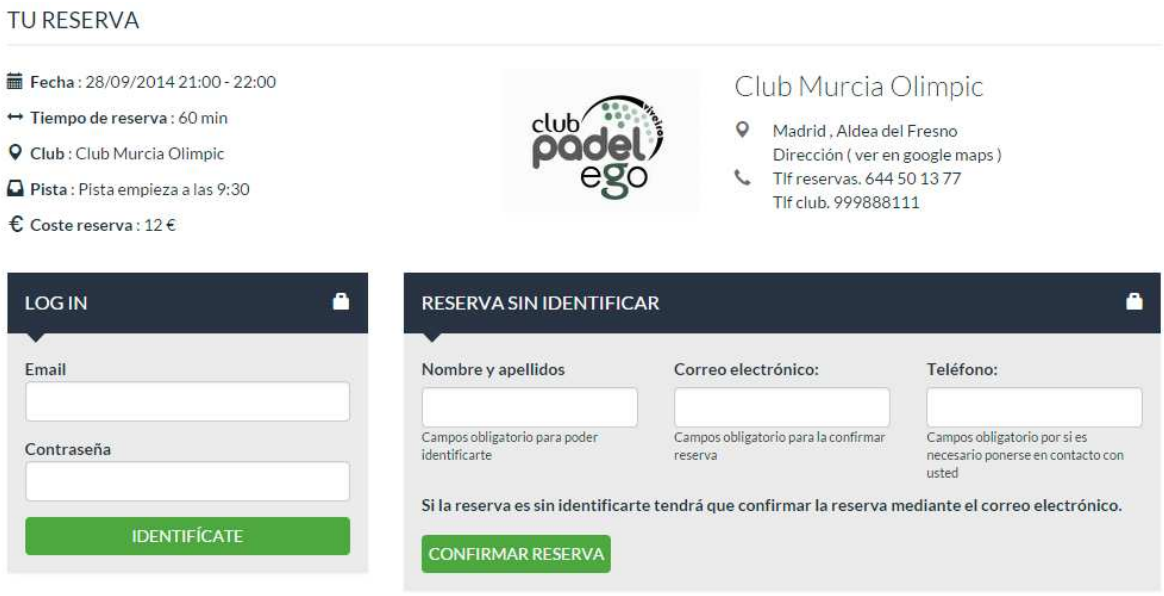
```
// LISTADO Y BUSQUEDA DE CLUB
'buscador' => array(
    'type' => 'Segment',
    'options' => array(
        'route' => '/buscador-pistas-de-padel[/:provincia[/:poblacion[/:slug]]]'
        'defaults' => array(
            'controller' => 'Application\Controller\Clubs',
            'action' => 'buscador',

        ),
    ),
```

El controlador asociado a este botón se encuentra en Application\Controller\Clubs. Ejecutará el método *buscadorAction()* localizado en *ClubsController.php*. Dicho método devuelve a la vista *buscador.phtml* el listado de clubs con pistas disponibles.

El usuario elegirá el club donde desea realizar la reserva. Tras pulsar el botón RESERVAR PISTA el servidor mapeará en el archivo *module.config.php* la ruta y ejecutará el controlador *ReservasController.php*. Éste contiene el método *indexAction()* que devolverá el horario de cada una de las pistas que tiene el club con una leyenda de disponibilidad.



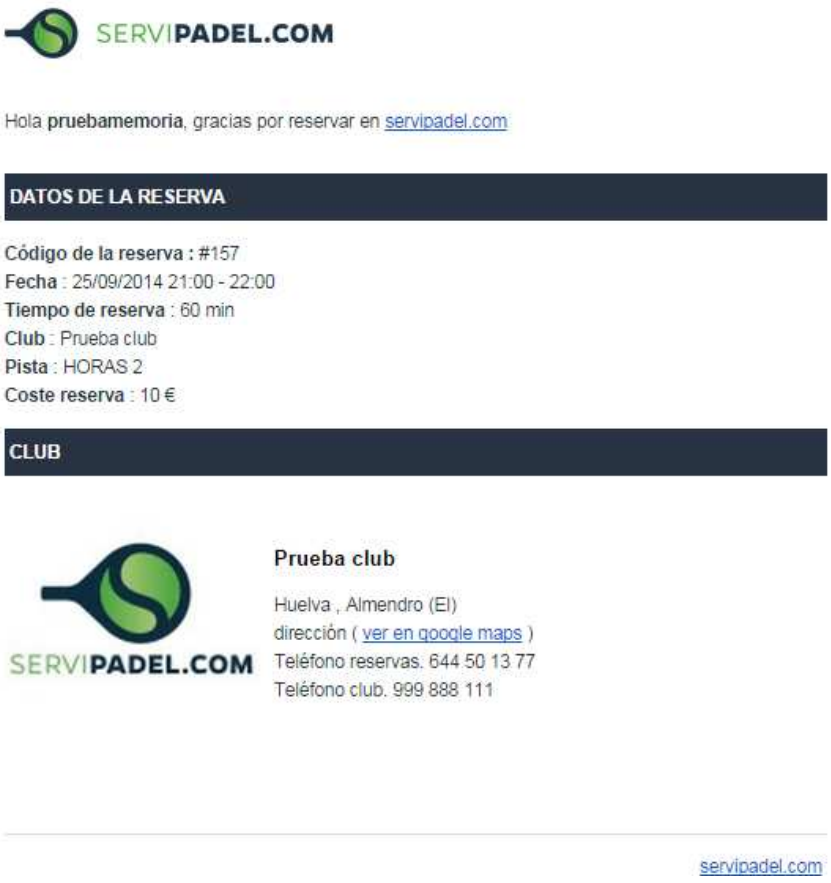
Si se pulsa el botón CONFIRMAR RESERVA ejecutará el método *confirmarReservaAction()*, alojado en Application/Controller/ReservasController.php. Éste método devolverá al usuario el formulario para confirmar la reserva y permitirá loguearse si es usuario de servipadel o sin identificar introduciendo nombre, correo electrónico y número de teléfono.



Si el usuario no desea registrarse introducirá los datos y pulsará el botón CONFIRMAR RESERVA que ejecutará el método *reservaCorrectaAction()* situado en la misma localización que el anterior. Enviará un correo electrónico al usuario y al administrador del club con todos los datos de la reserva.

```
//// ENVIO CORREO AL ADMINISTRADOR DEL CLUB.
if ($club->getCorreo() != "") {
    $content = $this->renderer->render('application/reservas/email-datos-club', array("reserva" =>
    $html = new MimePart($content);
    $html->type = "text/html";
    $body = new MimeMessage();
    $body->setParts(array($html));

    $message->addTo($club->getCorreo())
            ->addFrom('reservas@servipadel.com')
            ->setSubject('Nueva reserva de pista en servipadel.com')
            ->setBody($body);
    $transport->send($message);
}

//// ENVIO CORREO AL ADMINISTRADOR DEL CLUB.
$content = $this->renderer->render('application/reservas/email-datos', array("reserva" => $reserva,

$html = new MimePart($content);
$html->type = "text/html";
$body = new MimeMessage();
$body->setParts(array($html));

$message->addTo($reserva->getCorreo())
        ->addFrom('reservas@servipadel.com')
        ->setSubject('servipadel.com - Reserva realizada correctamente')
        ->addBcc("servipadel@servipadel.com")
        ->setBody($body);
$transport->send($message);
```

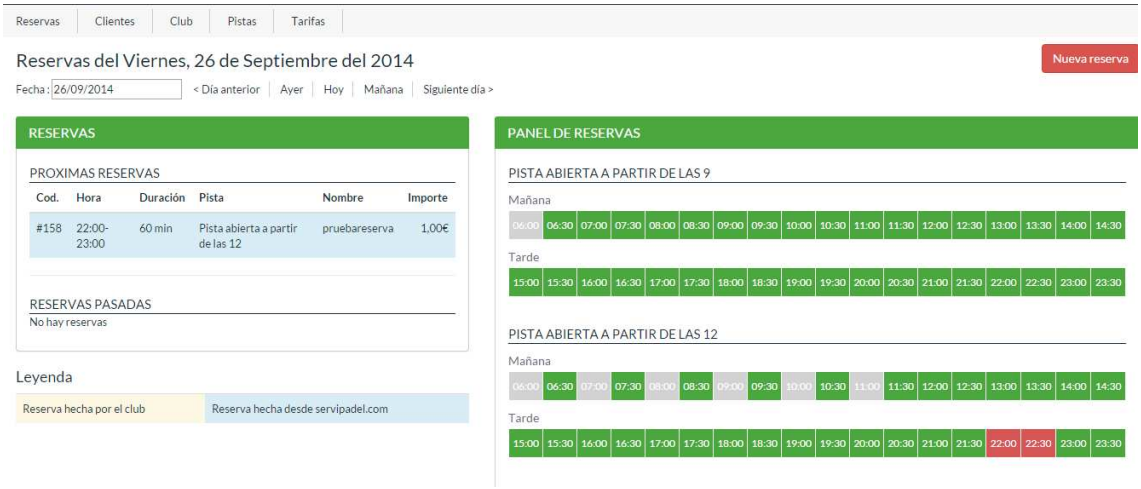
Ejemplo de correo enviado al usuario con los datos de la reserva realizada:



La reserva habrá finalizado dando al usuario las opciones de ir a inicio, registrarse, compartir en facebook o compartir en twitter. Si el usuario deseara compartir en facebook se ejecutaría en javascript el código que abrirá un nuevo explorador y le ofrecerá al usuario la posibilidad de compartir en su muro el enlace de servipadel.

5.4 Administrador de un club.

Si el usuario registrado es identificado como club el explorador nos mostrará por defecto el panel de reservas, con un menú en la parte superior para su gestión.



La URL de la petición es http://localhost/servipadel/gestion-club/reservas . La ruta se encuentra mapeada al controlador IndexController.php del módulo Club. Se ejecuta el método indexAction() que mostrará la vista anteriormente descrita.

```
'router' => array(
    'routes' => array(
        'gestionClub' => array(
            'type' => 'Segment',
            'options' => array(
                // Change this to something specific to your module
                'route' => '/gestion-club[/:controller[/:action[/:id]]]',
                'defaults' => array(
                    // Change this value to reflect the namespace in which
                    // the controllers for your module are found
                    '__NAMESPACE__' => 'Club\Controller',
                    'controller' => 'Index',
                    'action' => 'index',
                ),
            ),
```

Todas las acciones permitidas al club están disponibles en el menú. Cada opción del menú se redirecciona a los controladores correspondientes. A continuación se describen las funciones más interesantes que se pueden realizar por parte del administrador del club:

1. ReservasController.php:
    a. public function indexAction(): muestra el panel de reservas del club
    b. public function nuevaAction(): nueva reserva realizada por el club.
    c. public function infoReservaAction(): si el usuario hace click sobre alguna de las reservas, muestra toda la información de la reserva.

    d. public function cancelarReservaAction(): muestra la opción de cancelar la reserva si está seguro de ello
    e. public function cancelarReservaAction(): elimina la reserva si se ha cancelado.

2. ClientesController.php:
    a. public function indexAction(): muestra el listado de usuarios del club y un formulario para dar de alta a los clientes.
    b. public function editarAction(): posibilidad de editar los datos de los clientes.
    c. public function eliminarAction(): elimina el cliente.
    d. public function buscadorAction(): facilita la toma de datos de los usuarios ya dados de alta en la tabla de la base de datos clientes_club.

3. ClubController.php:
    a. editarAction(): el club podrá actualizar la información que se muestra de su club.

4. PistasController.php:
    a. public function indexAction(): lista las pistas creadas por el club.
    b. public function nuevaAction(): configuración de la nueva pista.
    c. public function duplicarAction(): duplica la pista seleccionada para que sea más sencillo su configuración.
    d. public function editarAction(): edita la configuración de la pista.

5. TarifasController.php:
    a. public function indexAction(): lista las tarifas creadas.
    b. public function nuevaAction(): crea una nueva tarifa.
    c. public function editarAction(): edita la configuración de la tarifa.
    d. public function eliminarAction(): borra la tarifa.

5.5 Administrador del portal web.

La vista mostrada al administrador es un menú donde se permite gestionar los clubs, usuarios y otros aspectos necesarios para mantener la web actualizada según necesidades del administrador.

El modelo que se ha creado para este usuario es *Admin*, compuesto por los controladores y las siguientes funciones más importantes:

1. ClubsController.php:
    a. public function indexAction(): lista los clubs creados.
    b. public function nuevoAction(): crea un nuevo club.
    c. public function editarAction(): modifica la información del club.
    d. public function eliminarAction(): elimina el club.

2. PistasController.php:
    a. public function indexAction(): lista todas las pistas creadas por los clubs.
    b. public function nuevaAction(): crea una pista nueva.
    c. public function editarAction(): modifica la configuración de la pista.
    d. public function eliminarAction(): borra la pista.

3. TarifasController.php:
    a. public function indexAction(): lista las tarifas creadas.
    b. public function nuevaAction(): crea una tarifa nueva
    c. public function editarAction(): modifica la configuración de la tarifa.
    d. public function eliminarAction(): elimina la tarifa.

4. InstalacionesController.php:
    a. public function indexAction(): lista las instalaciones creadas.
    b. public function nuevaAction(): crea una nueva instalación.
    c. public function editarAction(): modifica la configuración de la población.

5. ProvinciasController.php:
    a. public function indexAction(): lista las provincias creadas.
    b. public function nuevaAction(): crea una nueva provincia
    c. public function editarAction(): modifica la configuración de la provincia.
    d. public function eliminarAction(): elimina la provincia.

6. PoblacionesController.php:
    a. public function indexAction(): lista las poblaciones creadas.
    b. public function nuevaAction(): crea una nueva población.
    c. public function editarAction(): modifica la configuración de la población.
    d. public function eliminarAction(): elimina la población.

7. UsuariosController.php:
    a. public function indexAction(): lista las poblaciones creadas.
    b. public function editarAction(): modifica la configuración de la población.

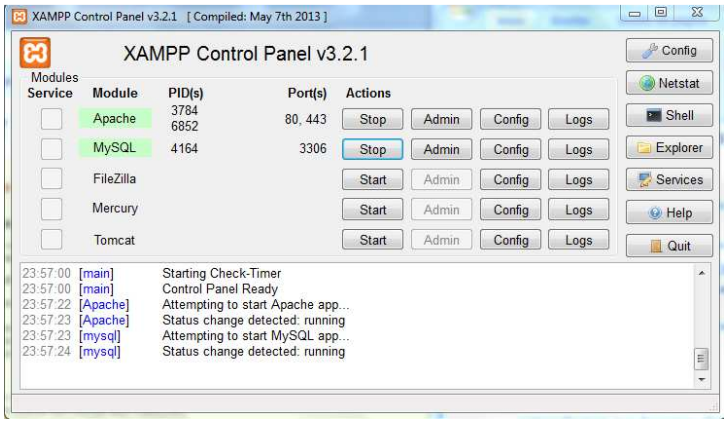
Apéndice.

INSTALACIÓN Y PUESTA EN MARCHA.

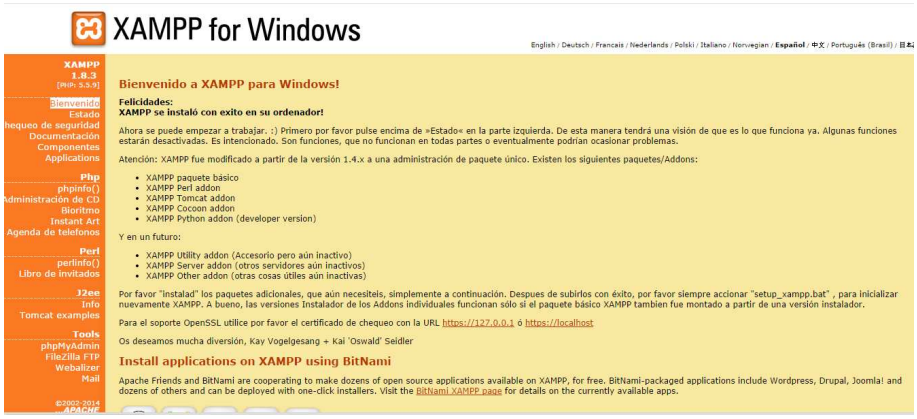
1. Instalación y configuración de XAMPP

Se debe descargar el paquete de instalación desde la dirección https://www.apachefriends.org/es/. Una vez descargado es fácil de instalar, únicamente ejecutamos el paquete descargado y debemos instalarlo en C:\xampp.

Una vez instalado arrancamos los servicios de Apache y MySQL en el panel de control de XAMPP.



Comprobamos que funciona abriendo un explorador y escribiendo en la barra de direcciones http://localhost/xampp. Si este enlace abre la siguiente vista habremos instalado correctamente el servidor HTTP Apache, el motor de la base de datos MySQL y PHP.

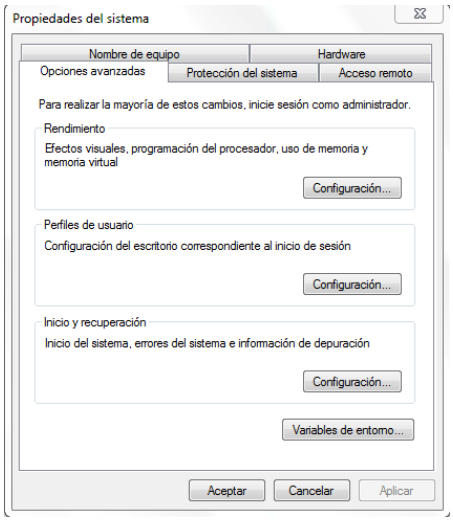


2. Instalación y configuración de Zend Framework 2.

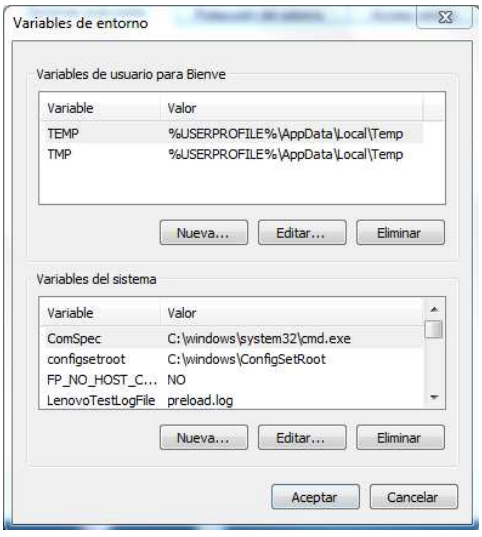
El primer paso es descargar la aplicación Zend Skeleton desde la dirección https://github.com/zendframework/ZendSkeletonApplication. Una vez descomprimida en la ruta C:/xampp/htdocs/zf2/ se instalará el archivo *composer.phar* con el que instalaremos las dependencias de ZF2 para poder correr la aplicación. Se ejecutará el comando *php composer.phar self-update* en una consola (símbolo de sistema) para actualizar a la versión más reciente y el comando php composer.phar install para instalar las dependencias.

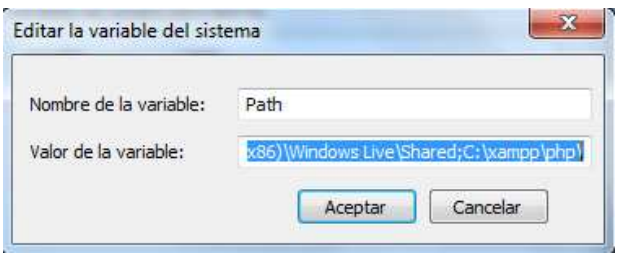
Si no se reconoce el comando php en la consola se deberán realizar los siguientes pasos y volver a actualizar el composer.phar:

1. Vamos a las Propiedades de Mi PC.
2. Seleccionamos la pestaña Opciones Avanzadas y luego click en "Variables de Entorno".



3. Dentro del cuadro Variables del Sistema buscamos la variable Path. En esta variable encontraremos una serie de path, lo que haremos es agregar con un punto y coma el path de nuestro archivo "php.exe" a esa variable.
   Seleccionamos la variable Path y con hacemos click en Modifica.
   Y a la lista de paths, SIN ELIMINAR ALGO, agregamos el path de php.exe *";C:\xampp\php\".*





En el directorio /vendor/zendframework/library/Zend se encuentran las librerías de ZF2.

Para comprobar su correcto funcionamiento se escribirá en la dirección del explorador http://localhost/zf2 mostrando la página de inicio de ZF2.

Bibliografía.

- Página oficial de PHP
  http://es.php.net/manual/es/

- PHP 6. Curso profesional de programación
  Edgar D'Andrea
  Ediciones InforBooks

- Página oficial Zend Framework
  http://zend.framework.com

- Videotutorial ZF2
  http://www.cesarcancino.com/categorias/detalle/zend-framework

- Desarrollo Web
  www.desarrolloweb.com