UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Aplicación Web para la gestión de un directorio de psicólogos usando framework CakePHP

Proyecto Final de Carrera

Ingeniería Técnica en Informática de Gestión

Autor: Javier Limorti Climent

Tutor: José Vicente Busquets Mataix

Curso 2014/2015

Resumen

Con el presente proyecto, se pretende diseñar una página web con un directorio de psicólogos donde las personas puedan encontrar la información más relevante del profesional que más le interese. Sea por su área geográfica, por los tratamientos que ofrece o por sus tarifas.

Se ofrecerá un sistema de contacto para comunicarse entre ellos a través de la web y un sistema de valoraciones en las que podrán puntuar y opinar sobre el psicólogo.

De forma interna, la web incluirá una área de administración en la que los administradores podrán gestionar toda la información relativa a los profesionales. Además, los psicólogos podrán acceder a un espacio dentro de la web donde podrán descargar todas las facturas por los servicios del portal. También tendrán la opción de efectuar el pago de la factura mediante Tarjeta de crédito/débito, Paypal[1] o Domiciliación bancaria.

Se utilizará el framework[2] CakePHP[3] para la realización del proyecto, aprovechando todo su potencial para crear la web y el área de administración.

Palabras clave: directorio, psicólogos, CakePHP, Aplicación web

1 https://www.paypal.com

2 https://es.wikipedia.org/wiki/Framework

3 http://cakephp.org/

Tabla de contenidos

1 Introducción

El principal objetivo del siguiente Proyecto Final de Carrera es ofrecer a la empresa en la que me encuentro actualmente, un modo de gestionar la información de todos sus clientes.

El proyecto se trata de una aplicación web que contendrá información de psicólogos y ofrecerá a las personas que la visiten un modo rápido de contacto entre ellos para solicitar cita en la consulta.

La empresa está especializada en la creación de páginas web para psicólogos en todo el ámbito nacional. Dispone de clientes con los que este proyecto podría lanzarse de forma inmediata al mercado, beneficiando a los antiguos clientes y ofreciendo a la empresa un modo de conseguir nuevos psicólogos que quieran obtener más pacientes en su consulta.

Por la necesidad de la empresa de un software de gestión y dado mi interés en utilizar nuevas tecnologías de programación he tomado la decisión de utilizar el lenguaje PHP[4] con el framework CakePHP y aprovechar todo su potencial para realizar este proyecto.

La página web tendrá tres áreas funcionales:

- El área principal donde toda aquella persona podrá consultar la información de los psicólogos disponibles en la base de datos.
- Una área donde los psicólogos dispondrán de un apartado donde visualizar las facturas por los servicios prestados y realizar el pago por ellos.
- Y otra área donde los administradores podrán realizar cambios en la información de los psicólogos.

2 Análisis y Diseño

2.1 Introducción

En este apartado serán definidas detalladamente las funcionalidades que deberá cumplir el proyecto para posteriormente realizar el correcto diseño de la aplicación satisfaciendo las necesidades de la empresa.

2.2 Requisitos específicos

- Existirán tres áreas funcionales como se comenta en la introducción. La portada de la web contendrá el buscador de psicólogos para todas aquellas personas que visiten la web. Una área privada para los psicólogos donde podrán descargar las

4 https://es.wikipedia.org/wiki/PHP

facturas y realizar el pago. Y finalmente, una área privada donde los administradores podrán administrar la web.

- Será necesario restringir el acceso a las diferentes áreas de la web utilizando un sistema de acceso con usuario y contraseña teniendo en cuenta que puede existir más de un usuario en cada área. Se debe cifrar la contraseña en la base de datos para mantener la privacidad de los usuarios.
- El buscador de psicólogos contendrá información relativa de cada profesional. Donde cualquier persona se podrá poner en contacto con un psicólogo para concertar una cita. Debe existir un método de entrada donde filtrar la búsqueda por la ciudad del psicólogo.
- Se incluirá un sistema de comentarios donde cualquier persona podrá valorar de forma pública el trato recibido por el psicólogo. Además, podrá puntuar el profesional con una escala del 1 al 5. De este modo, el psicólogo mejorará su reputación consiguiendo que más personas lleguen a su consulta.
- En la ficha personal del psicólogo deberá aparecer una fotografía suya para atraer al público y que sientan una mayor confianza para solicitar una cita.
- El área privada de los psicólogos contendrá un listado de las facturas emitidas por los servicios prestados. Se podrán descargar en versión PDF manteniendo el diseño actual de la empresa para las facturas, creándose de forma automática con los datos del cliente. Además, se debe ofrecer la posibilidad del pago de cada factura mediante Tarjeta de crédito/débito, Paypal o Domiciliación bancaria.
- Existirán varios tipos de contratos por las que un psicólogo podrá aparecer en la web. Y debe ser posible incluir nuevos tipos de contrato desde la parte administrativa.
- Será posible aparecer en el directorio de psicólogos tanto si ya tiene una página web como si la empresa le ofrece una.
- La información de cada psicólogo debe estar clasificada por la ciudad de origen, la provincia y el país al que pertenece.
- Los dominios tendrán la URL[5] de la web, un campo de texto donde anotar la descripción de la web y otros campos para ayudar a la empresa en la gestión de los dominios que son de su propiedad. En estos, se desea registrar si una web se encuentra posicionada en los buscadores o si está disponible para buscar un nuevo psicólogo para la web.
- Los contratos reflejarán la relación entre los dominios disponibles y los usuarios (psicólogos) de la web. Deberá contener la fecha en la que se creó el contrato y la fecha de finalización. Dar de baja un contrato debe ser tan fácil como marcar el estado del contrato como desactivado sin ser necesario eliminar sus datos.

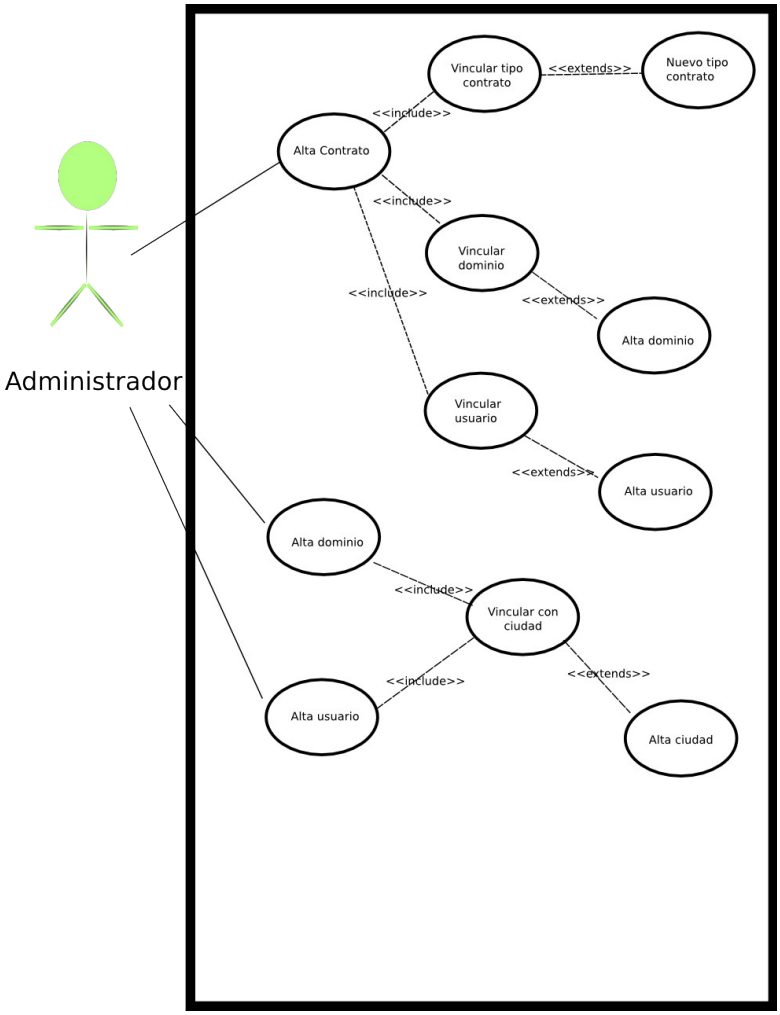
5 https://es.wikipedia.org/wiki/Localizador_de_recursos_uniforme

- El área de administración deberá tener todo lo relativo al mantenimiento de la información de la web, con los métodos de Alta, Editar, Eliminar y Ver de los dominios, contratos, usuarios, facturas, etc.
- El proyecto será diseñado siguiendo las tendencias actuales, siendo compatible con los dispositivos móviles (Responsive[6]). Además, debe incluir información de interés para otros psicólogos que quieran aparecer en la web y un método de contacto con la empresa.

2.3 Diagramas de casos de uso

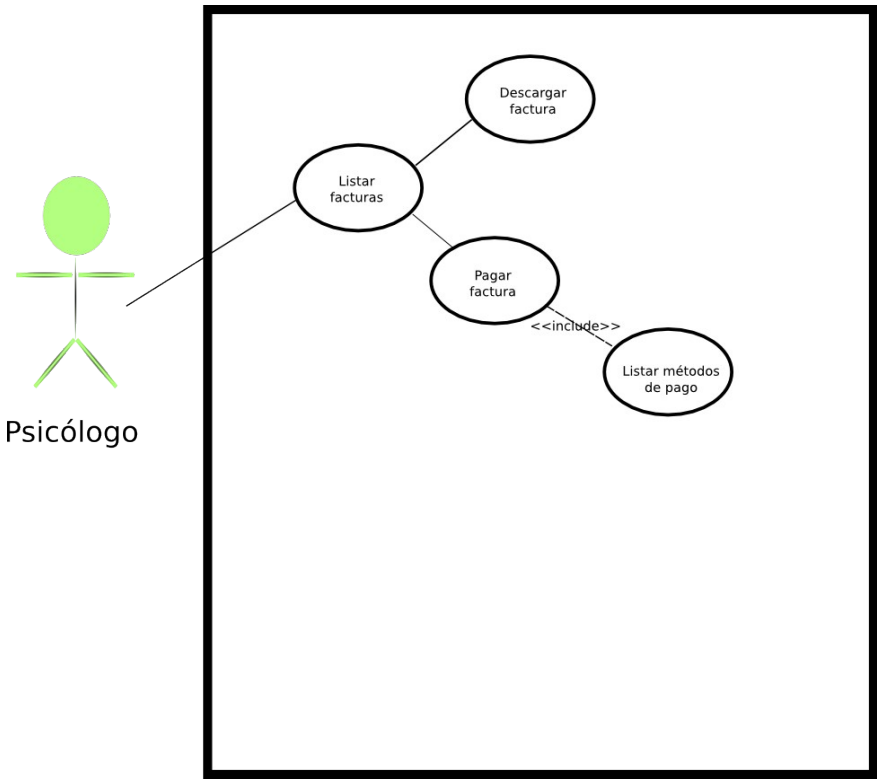
El diseño de los diagramas de casos de uso están realizados con el software libre de edición gráficos vectoriales Inkscape

A continuación, se muestran algunos de los diagramas realizados para el análisis de las acciones de la aplicación.

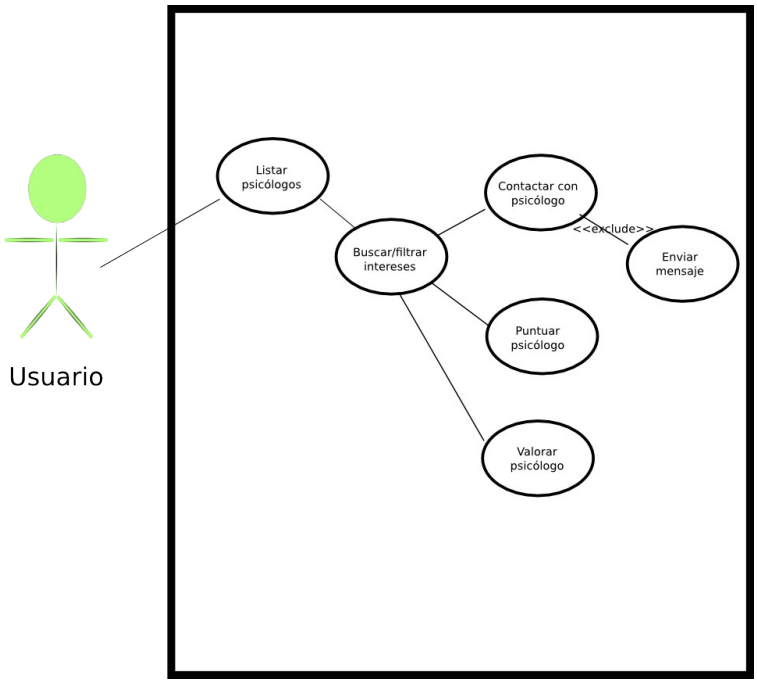


Dar de alta un contrato, un dominio y un usuario.

6 http://es.wikipedia.org/wiki/Diseño_web_adaptable



Psicólogo: Listar y pagar una factura



Usuario: Buscar a psicólogo; contacto, valoración y puntuación

2.4 Diagramas de clase

El diagrama de clases se ha creado utilizando el software libre MySQL Workbench 6.2[7] que ofrece la posibilidad de realizar el diseño del diagrama de clases de forma visual.

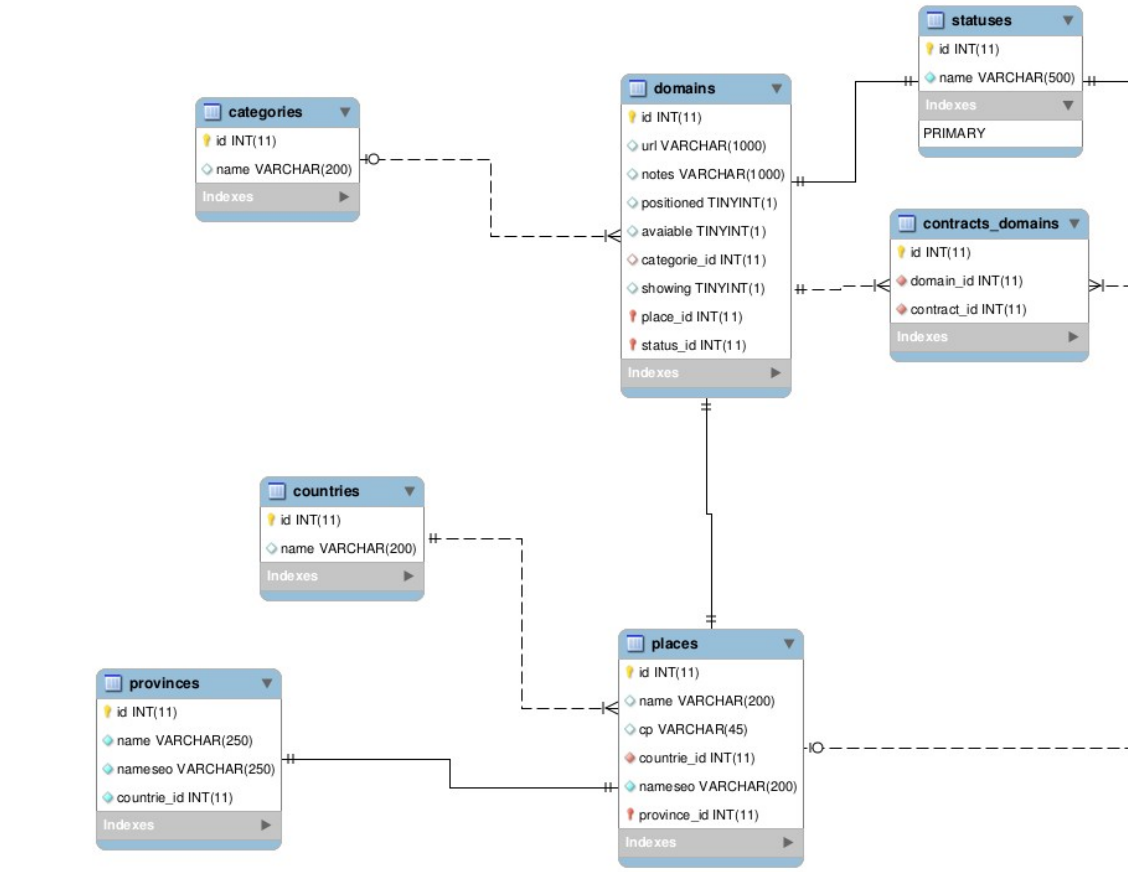
Es necesario mencionar que en CakePHP existe una convención[8] para los modelos de la base de datos que posteriormente facilitaran la creación de nuestra aplicación.

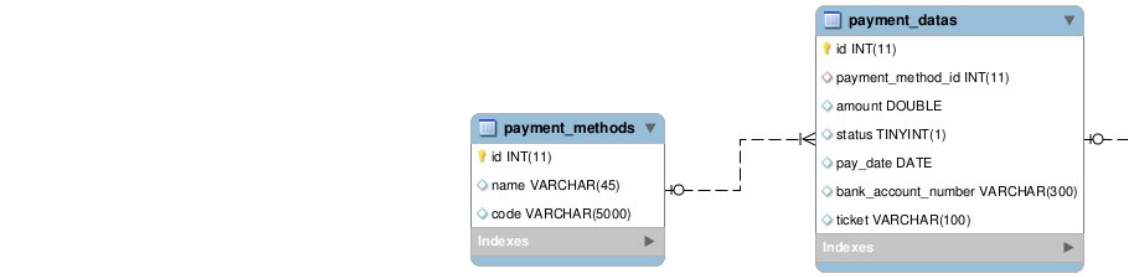
Entre ellas, destacar que los nombres de los modelos deben ir en plural como por ejemplo: el modelo Users. Y que todo modelo que tenga como clave foránea una relación a la tabla Users debe contener un campo llamado “user_id”.

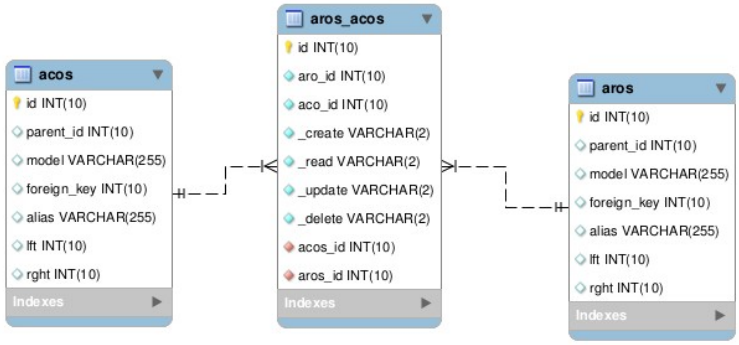
De este modo, el sistema reconocerá las relaciones construyendo una aplicación básica que se explicará detalladamente más tarde.

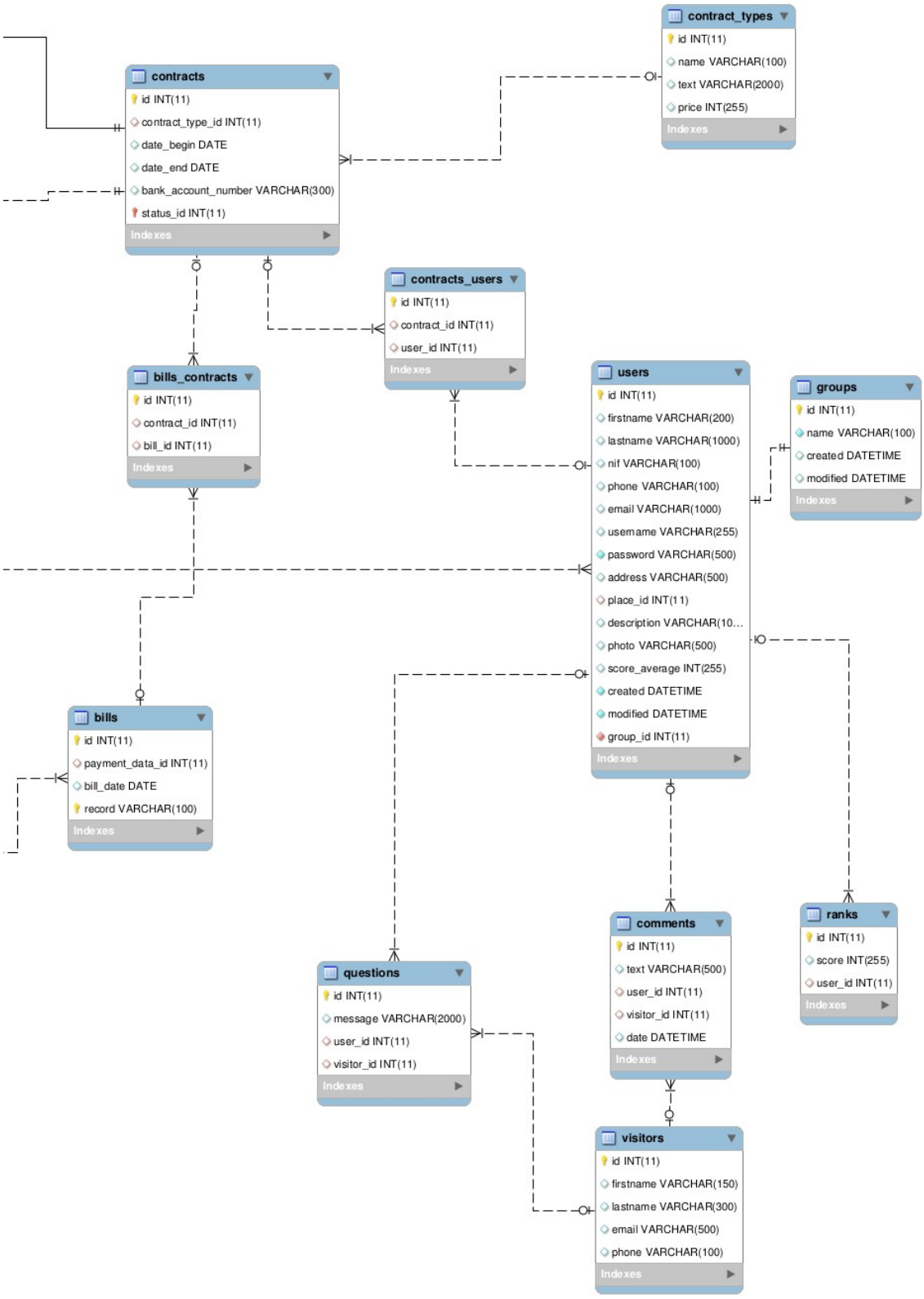
7 https://www.mysql.com/products/workbench/

8 http://book.cakephp.org/2.0/es/getting-started/cakephp-conventions.html


statuses
id INT(11)
name VARCHAR(500)
Indexes
PRIMARY
categories
id INT(11)
name VARCHAR(200)
Indexes
domains
id INT(11)
url VARCHAR(1000)
notes VARCHAR(1000)
positioned TINYINT(1)
avaiable TINYINT(1)
categorie_id INT(11)
showing TINYINT(1)
place_id INT(11)
status_id INT(11)
Indexes
contracts_domains
id INT(11)
domain_id INT(11)
contract_id INT(11)
Indexes
countries
id INT(11)
name VARCHAR(200)
Indexes
places
id INT(11)
name VARCHAR(200)
cp VARCHAR(45)
countrie_id INT(11)
nameseo VARCHAR(200)
province_id INT(11)
Indexes
provinces
id INT(11)
name VARCHAR(250)
nameseo VARCHAR(250)
countrie_id INT(11)
Indexes


payment_datas
id INT(11)
payment_method_id INT(11)
amount DOUBLE
status TINYINT(1)
pay_date DATE
bank_account_number VARCHAR(300)
ticket VARCHAR(100)
Indexes
payment_methods
id INT(11)
name VARCHAR(45)
code VARCHAR(5000)
Indexes


aros_acos
id INT(10)
aro_id INT(10)
aco_id INT(10)
_create VARCHAR(2)
_read VARCHAR(2)
_update VARCHAR(2)
_delete VARCHAR(2)
acos_id INT(10)
aros_id INT(10)
Indexes
acos
id INT(10)
parent_id INT(10)
model VARCHAR(255)
foreign_key INT(10)
alias VARCHAR(255)
lft INT(10)
rght INT(10)
Indexes
aros
id INT(10)
parent_id INT(10)
model VARCHAR(255)
foreign_key INT(10)
alias VARCHAR(255)
lft INT(10)
rght INT(10)
Indexes


contract_types
id INT(11)
name VARCHAR(100)
text VARCHAR(2000)
price INT(255)
Indexes
contracts
id INT(11)
contract_type_id INT(11)
date_begin DATE
date_end DATE
bank_account_number VARCHAR(300)
status_id INT(11)
Indexes
contracts_users
id INT(11)
contract_id INT(11)
user_id INT(11)
Indexes
bills_contracts
id INT(11)
contract_id INT(11)
bill_id INT(11)
Indexes
users
id INT(11)
firstname VARCHAR(200)
lastname VARCHAR(1000)
nif VARCHAR(100)
phone VARCHAR(100)
email VARCHAR(1000)
username VARCHAR(255)
password VARCHAR(500)
address VARCHAR(500)
place_id INT(11)
description VARCHAR(10...
photo VARCHAR(500)
score_average INT(255)
created DATETIME
modified DATETIME
group_id INT(11)
Indexes
groups
id INT(11)
name VARCHAR(100)
created DATETIME
modified DATETIME
Indexes
bills
id INT(11)
payment_data_id INT(11)
bill_date DATE
record VARCHAR(100)
Indexes
comments
id INT(11)
text VARCHAR(500)
user_id INT(11)
visitor_id INT(11)
date DATETIME
Indexes
ranks
id INT(11)
score INT(255)
user_id INT(11)
Indexes
questions
id INT(11)
message VARCHAR(2000)
user_id INT(11)
visitor_id INT(11)
Indexes
visitors
id INT(11)
firstname VARCHAR(150)
lastname VARCHAR(300)
email VARCHAR(500)
phone VARCHAR(100)
Indexes

2.5 Diseño

El framework CakePHP que se utiliza en el proyecto está basado en el patrón de diseño MVC (Modelo-Vista-Controlador). Con este patrón la estructura de la aplicación se separa en tres capas.

- El modelo representa la información del sistema almacenada en la base de datos para ser gestionada en la aplicación.
- La vista presenta la información obtenida de forma visual para el usuario.
- El controlador son las acciones que un usuario puede realizar en la aplicación. Sirve de intermediario entre la vista y el modelo para la obtención de los datos del modelo y servirlos en la vista.

Este patrón facilita el desarrollo de las aplicaciones y la evolución del software si en un futuro se requiere añadir nuevas funcionalidades. Por ejemplo, si en un futuro se necesita añadir alguna nueva funcionalidad en alguna de las entidades de nuestro modelo, será suficiente con añadir la acción al controlador y la vista correspondiente para representar la información en la aplicación.

Posteriormente, en la implementación se detallará cómo CakePHP hace uso del patrón de diseño MVC y como podemos generar todos los elementos de las tres capas de forma sencilla utilizando la linea de comandos.

3 Implementación

3.1 Preparación de CakePHP

Para tener preparado CakePHP para desarrollar nuestro proyecto, primero será necesario disponer de un servidor web donde alojar los archivos. Existen múltiples opciones para ello, pero en este proyecto se ha utilizado un servidor web Apache2[9], una base de datos MySQL 5.5[10], la versión 5 de PHP y el gestor de base de datos PhpMyAdmin[11] versión 4.2.12deb2 desde una distribución Debian[12] Jessie estable.

En la red se puede encontrar manuales para realizar la instalación del servidor y no se va a profundizar en ese aspecto.

La instalación ha sido en local, por lo que se deberá acceder a la aplicación web usando la url http://localhost

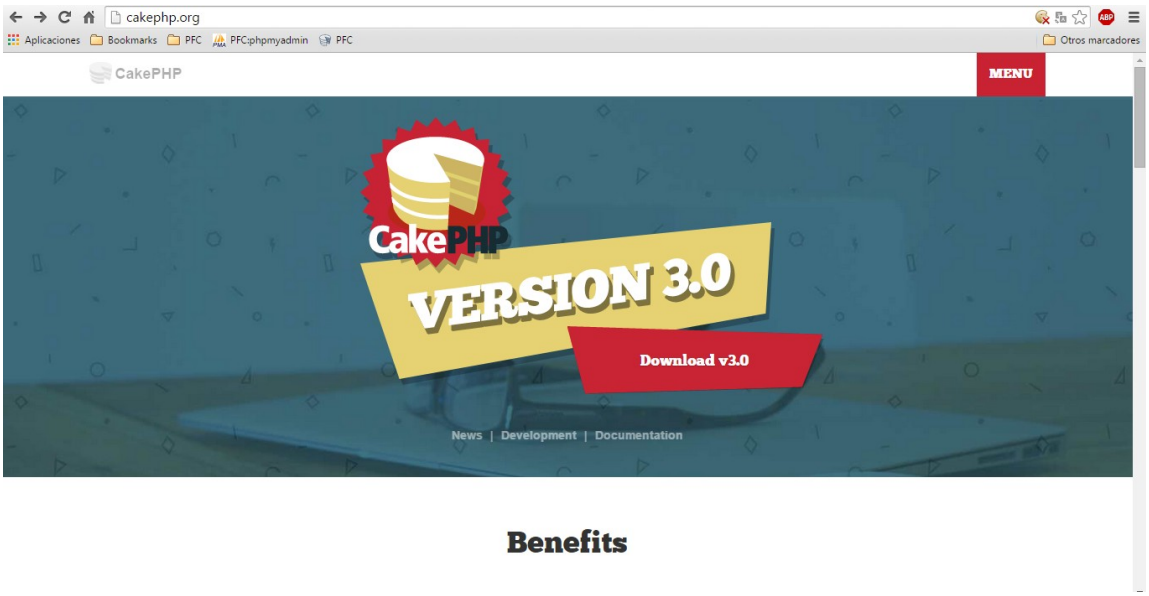
9 https://es.wikipedia.org/wiki/Servidor_HTTP_Apache

10 https://es.wikipedia.org/wiki/MySQL

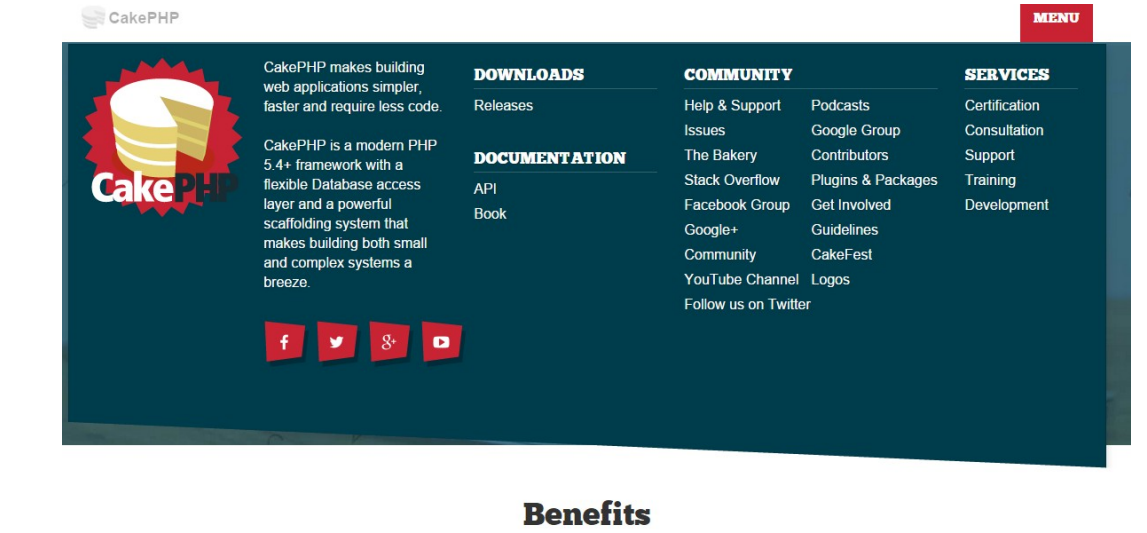
11 https://es.wikipedia.org/wiki/PhpMyAdmin

12 https://www.debian.org

Una vez el servidor está en funcionamiento, procederemos a acceder a la página web de CakePHP para descargar la versión del software que más nos interese.



Pulsamos en la opción “Releases” del Menú de la web:



Nos aparecerá un listado de archivos con las últimas versiones del proyecto. Deberemos descargar la que más nos interese. En este caso, se descargó la versión CakePHP v2.5.4.

Se seleccionó esta versión porque la documentación que encontraba se enfocaba en ella. En la última versión 3 de CakePHP se han introducido cambios y mejoras en el framework y descarté la opción por la posibilidad de no encontrar suficiente documentación para el desarrollo de este proyecto.

Una vez descargado el fichero zip, deberemos seguir las siguientes instrucciones para poder utilizar la opción Bake de CakePHP. Con el que podremos desarrollar nuestro proyecto de forma sencilla.

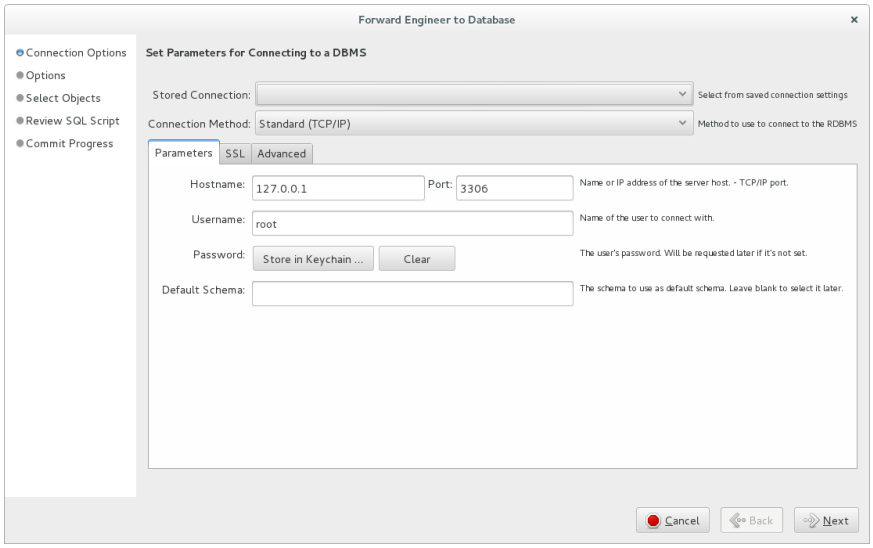
1. Descomprimimos el fichero:
   Unzip 2.5.4.zip
2. Al descomprimir el fichero, se creara una carpeta con el nombre cakephp-2.5.4. El contenido de esta carpeta debemos moverlo a la carpeta donde tenemos alojado el servidor.

   Mv cakephp-2.5.4/* /var/www/html/
3. A continuación, accedemos a /usr/bin
   cd /usr/bin
4. Por último, creamos un enlace simbólico para poder utilizar fácilmente Cake para generar nuestro proyecto.
   sudo ln -s /var/www/html/lib/Cake/Console/cake cake

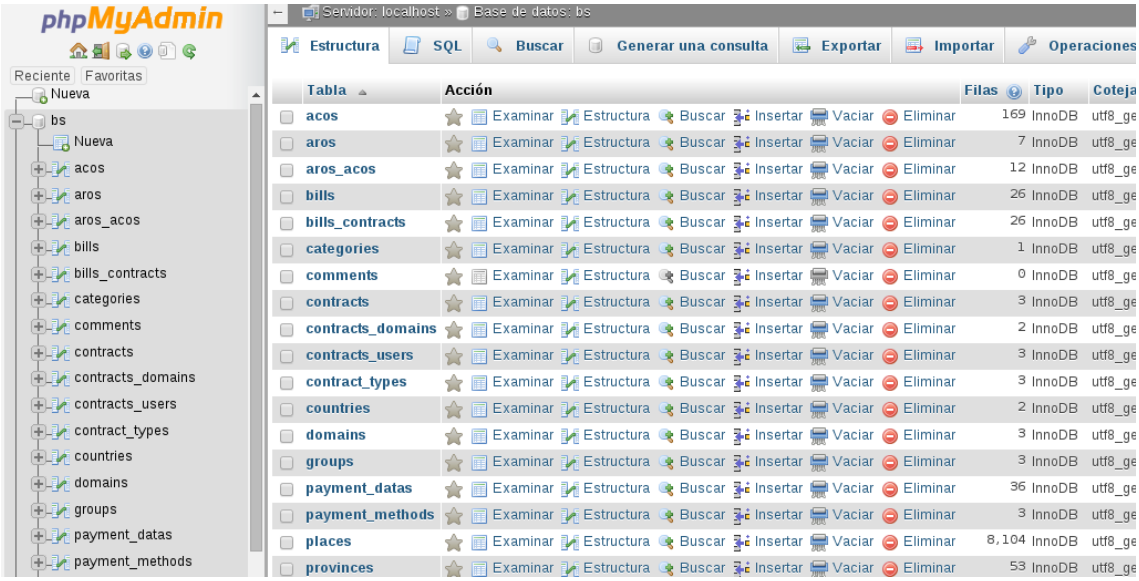
De este modo, podremos lanzar el comando cake desde la linea de comandos sin necesidad de especificar la ruta al fichero ejecutable.

3.2 Creación de las tablas en la base de datos

Mysql workbench ofrece la posibilidad de crear las tablas en el servidor MySQL de una forma muy sencilla con la opción “Forward Engineer” del menú “Database”. Conseguiremos que nuestro modelo sea volcado en la base de datos.



Con la aplicación web PhpMyAdmin para la gestión de bases de datos se puede observar como se ha realizado la creación de las tablas de nuestro modelo. Aunque también ofrece la posibilidad de copiar las instrucciones para crear las tablas con instrucciones SQL.



3.3 Uso de Bake en CakePHP

3.3.1 Conexión a la base de datos

Nos dirigimos a /var/www/html/ donde tenemos el proyecto. Y ejecutamos la orden “cake bake”. Nos preguntará donde queremos alojar nuestra aplicación. Por ejemplo, en /var/www/html/app

A continuación, nos realizará una serie de preguntas para configurar la conexión a nuestra base de datos que posteriormente podremos encontrar en el archivo database.php del directorio Config de nuestra aplicación.


```php
public $default = array(
        'datasource' => 'Database/Mysql',
        'persistent' => false,
        'host' => 'localhost',
        'login' => 'root',
        'password' => '*******',
        'database' => 'bs',
        'prefix' => '',
        'encoding' => 'utf8',
);
```

3.3.2 Modificaciones previas

Si se accede a la página web a través del navegador observamos como CakePHP nos advierte que será necesario modificar algunos parámetros de seguridad en el archivo app/Config/core.php

```
Configure::write('Security.salt', 'CADENA_PARA_MODIFICAR');

Configure::write('Security.cipherSeed','CADENA_NÚMERICA_MODIFICAR');
```

Los parámetros, son utilizados por CakePHP como semilla de las funciones Hash[13] y así incrementar la seguridad de nuestro sistema cuando se utilizan las funciones de cifrado.

Es posible, que CakePHP nos advierta de un fallo en el módulo Rewrite[14] del servidor web. Con este error, no nos será posible ver las imágenes y los estilos de nuestra web. Para solucionarlo se debe activar el módulo Rewrite de Apache lanzando la orden "a2enmod rewrite" con privilegios de root. Y modificar la configuración del servidor apache con las siguientes opciones:

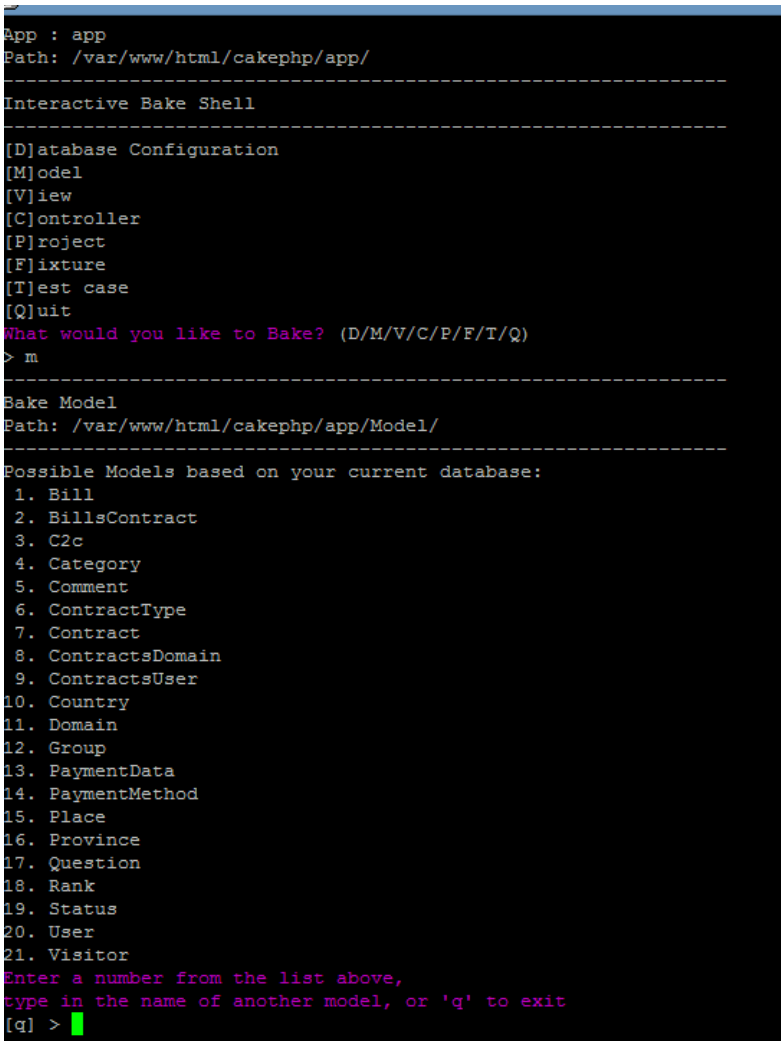
```
<Directory />
                Options FollowSymLinks
                AllowOverride All
</Directory>
<Directory /var/www/html>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride All
                Order Allow,Deny
                Allow from all
</Directory>
```

3.3.3 Creación de modelos

Si volvemos a ejecutar la orden "cake bake" desde nuestro directorio /var/www/html/ aparecerá un menú con distintas opciones. Podemos crear los modelos con unos sencillos pasos.

13 https://es.wikipedia.org/wiki/Función_hash

14 http://httpd.apache.org/docs/current/mod/mod_rewrite.html



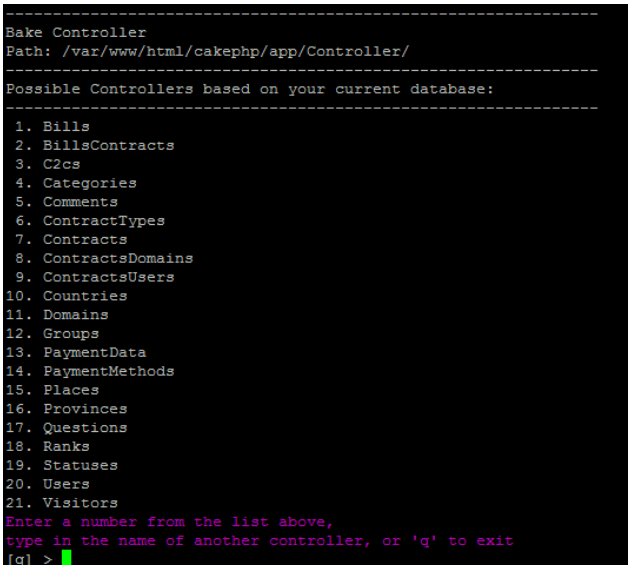
1. Nos pregunta sobre que Objeto queremos actuar. Recomiendo seguir el orden establecido (1-21) para generar todos los modelos y no dejar ningún modelo por crear. CakePHP creará los archivos PHP con los modelos de las tablas generando todo lo necesario.
2. Seleccionamos la primera opción: “Bill”.
3. Y nos pregunta: “A displayField could not be automatically detected would you like to choose one?”. En el caso de necesitarlo, deberemos especificar un campo de la tabla que será el dato mostrado en las relaciones. Usaremos “y” (yes) o “n” (no) para responder la pregunta y seleccionaremos el campo.
4. Un paso importante es “Would you like to supply validation criteria for the fields in your model?” con el que podemos especificar los requisitos para los campos. Por ejemplo, si se trata de un email podemos especificar que ese campo va a ser un email y CakePHP se encargará de la validación automática del campo.
5. “Would you like to define model associations?” analizará nuestra base de datos y si nuestro diseño está bien constituido realizará todas las relaciones existentes.

- Bill belongsTo PaymentData
- Bill hasAndBelongsToMany Contract

6. Por último, si estamos de acuerdo con la comprobación, podremos aceptar y guardar el modelo. Repetiremos los pasos del 1 al 6 para todas clases de nuestro proyecto.

3.3.4 Creación de controladores

Cómo se ha explicado anteriormente, volviendo al menú principal del bake existe una opción "controllers" para crear los archivos controladores de nuestro proyecto.



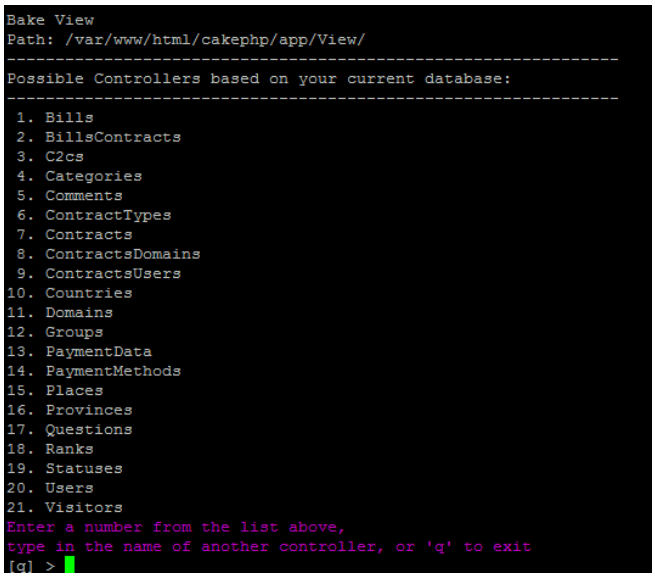
1. Deberemos seguir los pasos que nos indica y crear todos los controladores (1-21).
2. "Would you like to build your controller interactively?" Respondemos "yes" para generar con bake el controlador.
3. "Would you like to user dynamic scaffolding?" Respondemos "no" porque crearemos posteriormente las vistas de los controladores.
4. Would you like to create some basic class methods?" Le respondemos "yes", para que nos genere los métodos index, add, view y edit de nuestro objeto.
5. Would you like to create the basic class methods for admin routing?" No, usaremos ACL para generar los permisos del sistema resultando más facil.
6. Would you like this controller to use other helpers besides HtmlHelper and FormHelper? No
7. Would you like this controller to use other components besides PaginatorComponent? No, se utilizará el componente Pagination está muy

extendido y genera lo necesario para dividir los datos obtenidos en una consulta en páginas.

8. Would you like to use Session flash messages? Yes, estos mensajes se utilizan para mostrar mensajes de errores, y advertencias en la vista.

3.3.5 Creación de vistas

Y por último, seleccionamos la opción views del menú del bake. Para generar las visitas de los objetos de nuestro proyecto.



1. Would you like bake to build your views interactively? Warning: Choosing no will overwrite Bills views if it exist. Respondemos “Yes”.
2. Would you like to create some CRUD views (index, add, view, edit) for this controller? NOTE: Before doing so, you'll need to create your controller and model classes (including associated models). Respondemos “Yes” para generar las vistas por defecto.
3. Would you like to create the views for admin routing? Respondemos “No”.

3.4 Edición del código

3.4.1 Diseño responsive de la web.

En la actualidad, se ha extendido por toda la comunidad la tendencia a realizar las páginas web de modo Responsive. Se puede lograr que una página web adapte su contenido al tamaño del dispositivo que se está utilizando en ese momento para visualizar la web.

Dado que el propósito de este proyecto es realizar la gestión de la empresa y no necesariamente tener un diseño exclusivo, se ha seleccionado una plantilla de libre uso de la web html5up.net

Como se puede observar en la licencia de uso simplemente deberemos insertar una referencia en la web para reconocer el esfuerzo del autor.

En concreto, se ha utilizado la plantilla Miniport del catálogo de la web.

Para modificar la plantilla de nuestro proyecto por la descargada, se deberá mover los archivos con los estilos, las imágenes y los scripts en el directorio webroot.

Será necesario crear el fichero app/View/Layouts/bootstrap.ctp como se adjunta en la imagen de pantalla.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>
            <?php echo $title_for_layout; ?>
</title>
<meta name="viewport"
      content="width=device-width, initial-scale=1, maximum-scale=1">
<meta charset="utf-8">
<meta name="description" content="">
<meta name="author" content="">

      <?php
      echo $this->Html->meta ( 'icon' );
      echo $this->fetch ( 'meta' );
      echo $this->fetch ( 'css' );
      echo $this->fetch ( 'script' );
      echo $this->Html->css ( array (
                  'main',
```

```
                'bootstrap',
                'jquery.dataTables',
                'colorbox'
        ) );
        ?>
</head>
<body>
    <?php echo $this->Element('navigation'); ?>
    <div class="mainbody">
                <?php echo $this->Session->flash(); ?>
                <?php echo $this->fetch('content'); ?>
    </div>
      <!-- /.mainbody -->
      <?php
      echo $this->Html->script ( array (
                'jquery.min',
                'bootstrap',
                'jquery.scrolly.min.js',
                'skel.min',
                'skel-viewport.min',
                'util',
                'jquery.dataTables',
                'main',
                'jquery.colorbox',
                'initscripts'
        ) );
        ?>
  </body>
</html>
```

Es importante la etiqueta viewport para el correcto funcionamiento de la plantilla y que el navegador obtenga el ancho del dispositivo.

También se puede observar cómo se referencia desde el layout los archivos css y js que se van a utilizar desde la web con los parámetros:

```
$this->Html->css y $this->Html->script
```

Y cómo se puede obtener la barra de navegación de la aplicación que está almacenada en el fichero app/View/Elements/navigation.ctp

```
    <?php echo $this->Element('navigation'); ?>
```

Si deseamos que toda la aplicación utilice esta plantilla sólo será necesario editar el fichero app/Controller/AppController.php con el siguiente código en el método beforeFilter. Donde podemos configurar parámetros o realizar acciones que se ejecutarán antes de la generación de la vista.

```
    public function beforeFilter(){
      $this->layout = 'bootstrap';
    }
```

Finalmente, en el fichero app/View/Pages/home.ctp deberemos insertar el contenido de la página inicial de la aplicación. Podemos basarnos en la plantilla para estructurar la web utilizando los mismos elementos que tenia. Y posteriormente, modificar el contenido según nuestras necesidades.

3.4.2 Jquery

Jquery[15] es una libreria de javascript para facilitar la interacción entre los documentos HTML[16] y desarrollar código interpretado por el navegador. Actualmente es uno de los pilares para el desarrollo de las páginas web al permitir todo tipo funcionalidades.

La plantilla Miniport utiliza jQuery para dar soporte a los diferentes elementos que utiliza. Los componentes más destacables de la plantilla son los siguientes, aunque utiliza otros que no será necesario mencionar.

- Se utiliza bootstrap[17] para diseñar la web de modo Responsive. Facilita la maquetación de la página web con el catálogo de elementos de su biblioteca. Donde es posible especificar cómo debe verse una página html desde el navegador, desde un tablet o desde un teléfono móvil.
- Se utilizan fuentes de Font-Awesome[18] para insertar gráficos e iconos en la web. Estos gráficos son de libre uso y su utilización se encuentra muy extendida. Simplemente podemos insertar un gráfico con la forma de una casa introduciendo el siguiente código html. Estos símbolos son gráficos vectoriales que son interpretados por el navegador como un carácter del alfabeto.

15 https://jquery.com/

16 https://es.wikipedia.org/wiki/HTML

17 http://getbootstrap.com/

18 https://fortawesome.github.io/Font-Awesome/

```
<i class="fa fa-home"></i>
```

3.4.3 Datatables

Con los datatables[19] se puede crear una tabla con datos, un filtro y la paginación de forma muy sencilla cargando los datos mediante jQuery y Ajax[20].

Las llamadas Ajax son ejecutadas por el navegador cargando el resultado de la llamada en el documento html actual. Para el correcto funcionamiento, los datos cargados deben ser codificados en JSON[21] para que el componente datatable pueda realizar la carga correctamente.

Si la tabla es insertada dentro de un elemento DIV con la clase "table-responsive" bootstrap se encargará de que la tabla sea responsive.

La tabla aparentemente está vacía sirviendo de contenedor para los datos que se importaran mediante una llamada ajax en jQuery.

Se deberá especificar el nombre las columnas de la tabla respetando el número de columnas en el array en JSON.

```
<div class="table-responsive">
                                <table id="tablepsic" class="display"
cellspacing="0" width="100%">
                                        <thead>
                                              <tr>
                                                    <th>Name</th>
                                                    <th>Description</th>
                                                    <th>Photo</th>
                                                    <th>Url</th>
                                                    <th>Place</th>
                                                    <th>Actions</th>
                                              </tr>
                                        </thead>
                                        <tbody>
```

19 https://www.datatables.net/

20 http://api.jquery.com/jquery.ajax/

21 https://es.wikipedia.org/wiki/JSON

```
                                    </tbody>
                                    <tfoot>
                                          <tr>
                                                <th>Name</th>
                                                <th>Description</th>
                                                <th>Photo</th>
                                                <th>Url</th>
                                                <th>Place</th>
                                                <th>Actions</th>
                                          </tr>
                                    </tfoot>
                              </table>
                        </div>
```

Aquí lanzamos el script que cargará los datos recogidos por la acción /contracts/show y que será ejecutado en el momento que el documento se encuentre totalmente cargado.

```
$( document ).ready(function() {
      var oTable = $('#tablepsic').dataTable({
            bPaginate: true,
            sAjaxSource: "/contracts/show/",
            bSort: true,
      });
});
```

Estos datos deben ser codificados en el formato JSON para que el datatable pueda interpretar y cargar los datos en la tabla.

Si la cantidad de datos a recibir es excesivamente alta, se puede utilizar otro tipo de llamadas en ajax para hacer una carga parcial de los datos. Haciendo que el controlador gestione las llamadas para realizar la consulta en sql más concreta.

En este caso no se ha utilizado porque no se va a transmitir una gran cantidad de datos y además requiere una mayor elaboración porque se necesita incluir manualmente en el controlador el filtro y la paginación de la tabla.

A continuación se adjunta una imagen de pantalla del controlador donde podemos resaltar la opción recursive=2, para extender el array de resultados con las entidades directamente relacionadas con el modelo Contract.

```
//app/Controller/ContractsController.php
public function show() {
            $this->layout = 'json';
            $this->Contract->recursive = 2;
            $contracts = $this->Contract->find ( 'all' );
            $this->set ( 'contracts', $contracts );
            $this->set ( 'sEcho', count ( $contracts ) );
      }
```

Y la vista de la acción “show” del modelo “contracts” donde se codifica los datos de la consulta en formato JSON. Es importante que la salida del código sea codificada en JSON utilizando la función json_encode.

```
<?php
$output = array (
            "sEcho" => $sEcho,
            "iTotalRecords" => 0,
            "iTotalDisplayRecords" => 0,
            "aaData" => array ()
);

$counter = 0;
foreach ( $contracts as $contract ) {

      if ($contract ['User'] [0] ['group_id'] == 6) { // sólo se
muestra si pertenece al grupo de psicólogos
```

```
            $field1 = $contract ['User'] [0] ['firstname'] . " " .
$contract ['User'] [0] ['lastname'];

            $field2 = $contract ['User'] [0] ['description'];


            $fieldphoto = "<div class='datatable_photo'><img
src='/files/user/photo/" . $contract ['User'] [0] ['id'] . "/" .
$contract ['User'] [0] ['photo'] . "'>";

            if (! empty ( $contract ['Domain'] )) { // si no tiene web
asociada, mostramos el lugar de residencia del psicólogo y nada en el
campo web


                  $field3 = '<a href="http://' . $contract ['Domain']
[0] ['url'] . '">' . $contract ['Domain'] [0] ['url'] . '</a>';

                  $field4 = $contract ['Domain'] [0] ['Place']
['name'];

            } else {

                  $field3 = "-";

                  $field4 = $contract ['User'] [0] ['Place'] ['name'];

            }

            $field5 = '<a class="boxlink" href="/users/moreinfo/' .
$contract ['User'] [0] ['id'] . '"><span class="glyphicon glyphicon-
search"></span></a>';


            $output ["aaData"] [] = array (

                        $field1,

                        $field2,

                        $fieldphoto,

                        $field3,

                        $field4,

                        $field5

            )

            ;

            $counter ++;

      }

}
```

```
$output ['iTotalRecords'] = $counter; // suma de todos

$output ['iTotalDisplayRecords'] = $counter; // si utilizo search, los
filtrados en la consulta


echo json_encode ( $output );


?>
```

Finalmente, se adjunta una imagen de la tabla creada con datatables que se encuentra en la página inicial de la aplicación. Donde las personas podrán obtener la información de los psicólogos.

3.4.4 Colorbox

Con el componente Colorbox[22] se puede lanzar una subventana dentro de nuestra aplicación cargando la información en ella mediante Ajax. Podemos crear otro layout más simple para que esta subventana no contenga todos los elementos del layout "bootstrap". Por ejemplo, el menú de navegación de la web.

Volviendo al fichero initscripts.js, se observa el siguiente código para la carga de datos en la subventana. En este caso, el script se activa cuando un usuario hace click en un enlace de la web.

Al hacer click, el script recupera la url del elemento que ha sido invocado consultado el atributo href. El contenido de esta url es cargado mediante ajax en la subventana creada por colorbox.

22 http://www.jacklmoore.com/colorbox/

```
$(document).ajaxSuccess(function(e){
            $(".boxlink").click(function(){
                  $.colorbox({
                        href:$(this).attr('href'),
                        width:"95%",
                        height:"95%",
                   });
                  return false;
             });
```

Como se puede observar, el evento no se encuentra en el evento Ready sino en el evento ajaxSuccess. Con ello se ha solucionado un problema de compatibilidad entre el datatable y colorbox.

Previamente, los enlaces de dentro del datatable que debían ser abiertos por colorbox en una subventana no funcionaban. De este modo, el evento click para abrir la subventana se declara después de cargar por completo la tabla del datatable consiguiendo el comportamiento deseado.

3.4.5 ACL

Se utiliza el plugin ACL de Alaxos [23]para crear los permisos de los usuarios y los grupos de la aplicación.

Existen otros métodos para establecer los permisos con CakePHP pero se ha seleccionado este plugin por ser una de las opciones más documentadas en la

23 https://github.com/sams/alaxos_acl

comunidad. Donde los usuarios han compartido los obstáculos que han tenido en la instalación y la solución que han aplicado.

La instalación del plugin está documentada en la web y obtendremos un panel de administración donde modificar los permisos de los usuarios de forma muy visual.

Toda la información relativa a los permisos de los usuarios y los grupos es almacenada en las tablas AROS, ACOS y AROS_ACOS de la base de datos. Mientras que el modelo ACOS almacena la información de los controladores y las acciones de la aplicación. El modelo AROS establece la pertenencia de los usuarios a los distintos grupos de usuarios. El modelo que las relaciona AROS_ACOS establece los permisos de cada grupo para cada acción de nuestra aplicación.

3.4.5.1 Permisos generales

Anteriormente se ha descrito cómo se puede establecer los permisos para todas las acciones de cada grupo de usuario. Si deseamos establecer unas acciones que puedan utilizarse por toda aquella persona que no se identifique en la web, deberemos editar el método beforefilter en app/Controller/AppController.php con el siguiente parámetro.

```
$this->Auth->allow ( 'display', 'contact', 'moreinfo', 'show',
'value', 'ipn', 'tpv' );
```

Mientras se configura este componente podemos permitir todas las acciones de la aplicación a todos los usuarios.

```
$this->Auth->allow();
```

Esta opción sólo es recomendada mientras la aplicación no está en producción porque de lo contrario cualquier persona podría acceder al área de administración de la web obteniendo los datos de los clientes.

3.4.6 Seguridad de la contraseña

Si se desea aumentar la seguridad de la aplicación almacenando las contraseñas de los usuarios con un cifrado podemos incluir en el modelo "User" la siguiente configuración. Donde se especifica que antes de guardar un usuario, será necesario cifrar la contraseña utilizando el componente AuthComponent de CakePHP. En este momento se utilizará el Hash que hemos configurado anteriormente en el fichero core.php

```
      public function beforeSave($options = array()) {
            if (isset ( $this->data ['User'] ['password'] )) {
                  $this->data ['User'] ['password'] =
AuthComponent::password ( $this->data ['User'] ['password'] );
                  return true;
            }
      }
```

3.4.7 Plugin cakephp-upload

El modelo "User" contiene un campo "photo" donde un psicólogo podrá tener una fotografía suya para publicar en la web.

Se ha utilizado el plugin cakephp-upload[24] para conseguir subir una fotografía desde la acción "editar" del usuario.

Dispone una amplia documentación[25] donde se especifica los pasos a seguir para instalar y utilizar el plugin.

Finalmente se deberá modificar la vista para mostrar una imagen en html con la ruta al fichero anteriormente subido que se almacena en el directorio files del webroot.

El plugin genera un directorio "photo" por el campo del formulario y un subdirectorio para cada usuario con el id donde almacena la fotografía.

24 https://github.com/josegonzalez/cakephp-upload

25 http://cakephp-upload.readthedocs.org/en/latest/index.html

3.4.8 Creación de PDF para las facturas

Para la creación de facturas en PDF por los servicios de los psicólogos se utiliza la librería Fpdf[26]. En la siguiente imagen se observa como se prepara los datos desde el controlador, para procesarlos en la vista generando el PDF.

```
public function viewpdf($id = null) {
            $this->Bill->recursive = 4;
            if (! $this->Bill->exists ( $id )) {
                  throw new NotFoundException ( __ ( 'Invalid bill'
) );
            }
            $options = array (
                        'conditions' => array (
                                    'Bill.' . $this->Bill->primaryKey
=> $id
                        )
            );
            $bill = $this->Bill->find ( 'first', $options );

            $iduser = $this->Session->read ( 'Auth.User.id' );

            if ($this->Session->read ( 'Auth.User.Group.name' ) !=
'admin' && $iduser != $bill ['Contract'] [0] ['User'] [0] ['id']) {
                  // Sólo mostramos la factura en el caso de ser un
usuario administrador o que la factura sea del usuario actual.
                  return $this->flash ( __ ( 'The bill could not be
shown. Please, try again.' ), array (
                              'action' => 'showlist'
                  ) );
            }

            $this->set ( 'bill', $bill );

            // Import /app/Vendor/Fpdf
```

26 http://www.fpdf.org/

```
            App::import ( 'Vendor', 'Fpdf', array (
                        'file' => 'fpdf/fpdf.php'
            ) );
            // Assign layout to /app/View/Layout/pdf.ctp
            $this->layout = 'pdf'; // this will use the pdf.ctp layout
                                   // Set fpdf variable to use in view
            $this->set ( 'fpdf', new FPDF ( 'P', 'mm', 'A4' ) );

            // render the pdf view (app/View/[view_name]/pdf.ctp)
            $this->render ( 'pdf' );
      }
```

También será necesario destacar, la siguiente línea de código, ya que comprueba que la factura solamente pueda ser descargada por los administradores o por el usuario al que pertenece.

```
if ($this->Session->read('Auth.User.Group.name')!='admin' && $iduser!
=$bill['Contract'][0]['User'][0]['id'])
```

Por ello, se obtiene el identificador del usuario de la variable de sesión y el grupo al que pertenece.

Finalmente, en la vista correspondiente se realiza el diseño del documento pdf con los datos obtenidos.

```
$fpdf->AddPage ();

// logo de la tienda
$fpdf->Image ( WWW_ROOT . '/img/ama.jpg', 5, 5, 25, 20, 'JPG',
'http://www.amalapublicidad.com' );

// Encabezado de la factura
$fpdf->SetFont ( 'Arial', 'B', 14 );
$fpdf->Cell ( 190, 10, "FACTURA", 0, 2, "C" );
$fpdf->SetFont ( 'Arial', 'B', 10 );
// $numyear = explode("-",$bill['Bill']['bill_date']);
// $numyear = substr($numyear[0],2,4);
```

```
$fpdf->MultiCell ( 190, 5, utf8_decode ( "Factura nº: " ) . $bill
['Bill'] ['record'] . "\n" . "Fecha: " . $bill ['Bill'] ['bill_date'],
0, "C", false );

$fpdf->Ln ( 2 );


// Datos de la tienda

$fpdf->SetFont ( 'Arial', 'B', 12 );

$top_datos = 45;

$fpdf->SetXY ( 40, $top_datos );

$fpdf->Cell ( 190, 10, "AMAlapublicidad", 0, 2, "J" );

$fpdf->SetFont ( 'Arial', '', 9 );

$fpdf->MultiCell ( 190, // posición X

5, // posición Y

utf8_decode ( "****************\n" ) . utf8_decode ( "****\n" ) .
utf8_decode ( "Población: Valencia\n" ) . utf8_decode ( "Provincia:
Valencia\n" ) . utf8_decode ( "Código Postal: 46002\n" ) . utf8_decode
( "Teléfono: *****\n" ) . utf8_decode ( "N.I.F./C.I.F.: ******" ), 0,
// bordes 0 = no | 1 = si

"J", // texto justificado

false );


// Datos del cliente

$fpdf->SetFont ( 'Arial', 'B', 12 );

$fpdf->SetXY ( 125, $top_datos );

$fpdf->Cell ( 190, 10, "Datos del cliente", 0, 2, "J" );

$fpdf->SetFont ( 'Arial', '', 9 );

$fpdf->MultiCell ( 190, // posición X

5, // posicion Y

utf8_decode ( "Nombre: " . $bill ['Contract'] [0] ['User'] [0]
['firstname'] ) . "\n" . utf8_decode ( "Apellidos: " . $bill
['Contract'] [0] ['User'] [0] ['lastname'] ) . "\n" . utf8_decode (
"Dirección: " . $bill ['Contract'] [0] ['User'] [0] ['address'] ) .
"\n" . utf8_decode ( "Población: " . $bill ['Contract'] [0] ['User']
[0] ['Place'] ['name'] ) . "\n" . utf8_decode ( "Provincia: " . $bill
['Contract'] [0] ['User'] [0] ['Place'] ['Province'] ['name'] ) . "\n"
. utf8_decode ( "Código Postal: " . $bill ['Contract'] [0] ['User']
[0] ['Place'] ['cp'] ) . "\n" . utf8_decode ( "N.I.F./C.I.F.: " .
```

```
$bill ['Contract'] [0] ['User'] [0] ['nif'] ), 0, // bordes 0 = no | 1 = si

"J", // texto justificado

false );


// Salto de línea

$fpdf->Ln ( 2 );


$top_productos = 110;

$fpdf->SetXY ( 45, $top_productos );

$fpdf->Cell ( 40, 5, 'CONCEPTO', 0, 1, 'C' );

$fpdf->SetXY ( 115, $top_productos );

$fpdf->Cell ( 40, 5, 'IMPORTE (' . EURO . ')', 0, 1, 'C' );


$y = 115;

$x = 0;


// Cálculo del Impuesto

$TotalSinIVA = round ( $bill ['PaymentData'] ['amount'] / 1.21, 2 );

$IVA = $bill ['PaymentData'] ['amount'] - $TotalSinIVA;


$fpdf->SetFont ( 'Arial', '', 8 );


$fpdf->SetXY ( 45, $y );

$fpdf->Cell ( 40, 5, "Servicios de " . $bill ['Contract'] [0] ['Domain'] [0] ['url'], 0, 1, 'C' );


$fpdf->SetXY ( 115, $y );

$fpdf->Cell ( 40, 5, $TotalSinIVA . " " . EURO, 0, 1, 'C' );


$fpdf->SetXY ( 90, $y + 5 );

$fpdf->Cell ( 40, 5, "Base imponible:", 0, 1, 'C' );
```

```
$fpdf->SetXY ( 115, $y + 5 );
$fpdf->Cell ( 40, 5, $TotalSinIVA . " " . EURO, 0, 1, 'C' );


$fpdf->SetXY ( 90, $y + 10 );
$fpdf->Cell ( 40, 5, "I.V.A: 21%", 0, 1, 'C' );


$fpdf->SetXY ( 115, $y + 10 );
$fpdf->Cell ( 40, 5, $IVA . " " . EURO, 0, 1, 'C' );


$fpdf->SetXY ( 90, $y + 15 );
$fpdf->Cell ( 40, 5, "TOTAL: ", 0, 1, 'C' );


$fpdf->SetXY ( 115, $y + 15 );
$fpdf->Cell ( 40, 5, $bill ['PaymentData'] ['amount'] . " " . EURO, 0,
1, 'C' );


// logo de la tienda
$fpdf->Image ( WWW_ROOT . '/img/ama-opacity.jpg', 60, $y + 30, 70, 56,
'JPG', 'http://www.amalapublicidad.com' );


$textoLOPD = "TEXTO_DE_LA_LEY_DE_PROTECCIÓN_DE_DATOS";


$fpdf->SetXY ( 115, $y + 100 );
$fpdf->Ln ( 5 );
$fpdf->SetFont ( 'Arial', '', 8 );
$fpdf->Write ( 5, utf8_decode ( $textoLOPD ) );


$fpdf->Output ();
?>
```

3.4.9 Listado de facturas

Este listado se encuentra disponible para cualquier usuario, mostrándose solamente sus facturas. No tiene acceso a las facturas de otros usuarios.

Las de color rojo, que no han sido pagadas, tienen la opción de pagar mediante tarjeta de crédito/débito, paypal o domiciliación bancaria.

Las de color azul, son las facturas que ya han sido abonadas anteriormente por el usuario.

En la siguiente imagen se adjunta el código del controlador que obtiene los datos de las facturas. Es necesario destacar que se ha utilizado en la consulta la opción JOIN para obtener los datos de los contratos y los pagos, del usuario actualmente autenticado.

```
public function showlist() {
            $this->Bill->recursive = 4;


            $id = $this->Session->read ( 'Auth.User.id' );


            $options = array (
                        'joins' => array (
                                    array (
                                                'table' =>
'contracts_users',
                                                'alias' =>
'ContractsUsersJoin',
```

```
                                                'type' => 'INNER',
                                                'conditions' => array
(
                                                        'Contracts
UsersJoin.contract_id = Contract.id'
                                                )
                                        )
                        ),
                        'conditions' => array (
                                        'ContractsUsersJoin.user_id = "' .
$id . '"'
                        )
            )
            ;

            $contracts = $this->Bill->Contract->find ( 'all', $options
);

            $paymentdata = $this->Bill->PaymentData->find ( 'all' );

            $this->set ( 'contracts', $contracts );
            $this->set ( 'paymentdata', $paymentdata );
      }
```

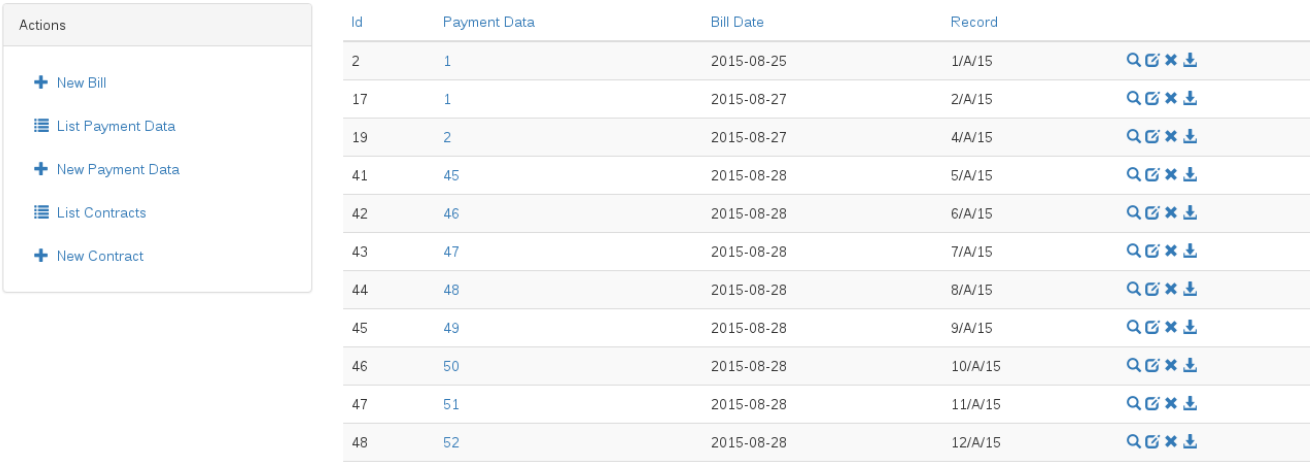
Y en la vista de la acción, podemos observar como primero obtenemos para cada factura (Bill) su linea de pago correspondiente de la entidad PaymentData.

```
foreach ( $contracts [0] ['Bill'] as $bill ) {
                            $position = - 1;
                            foreach ( $paymentdata as $key => $pd )
{
                                    if ($pd ['PaymentData'] ['id'] ==
$bill ['payment_data_id']) {
```

```
                                        $position = $key;
                                    }
                                }
                                if ($position >= 0) {
                                    if ($paymentdata [$position]
['PaymentData'] ['status'] == 1) {
                                        $statusclass = "paid";
                                        $statusbutton = "statebutton
fa-thumbs-up";
                                    } else {
                                        $statusclass = "nopaid";
                                        $statusbutton = "statebutton
fa-times-circle";
                                    }
                                    echo '<div class="listitem col-md-
2">';
                                    echo $this->Html->link ( h ( $bill
['record'] ) . '<br><i class="fa fa-file-pdf-o ' . $statusclass .
'"></i>', array (
                                                'action' => 'viewpdf',
                                                h ( $bill ['id'] )
                                    ), array (
                                                'escape' => false
                                    ) );

                                    echo '<i class="fa ' .
$statusbutton . '"></i>';

                                    if ($statusclass == "nopaid") {
                                        echo $this->Html->link ( '<i
class="fa fa-credit-card paybutton"></i>', array (
                                                    'controller' =>
'payment_data',
                                                    'action' =>
'pay',
                                                    'Tpv',
```

```
                                                    h ( $bill
['payment_data_id'] )
                                            ), array (
                                                    'escape' =>
false
                                            ) );
                                            echo $this->Html->link ( '<i
class="fa fa-cc-paypal paybutton"></i>', array (
                                                    'controller' =>
'payment_data',
                                                    'action' =>
'pay',
                                                    'Paypal',
                                                    h ( $bill
['payment_data_id'] )
                                            ), array (
                                                    'escape' =>
false
                                            ) );
                                            echo $this->Html->link ( '<i
class="fa fa-bank paybutton"></i>', array (
                                                    'controller' =>
'payment_data',
                                                    'action' =>
'pay',
                                                    'Sepa',
                                                    h ( $bill
['payment_data_id'] )
                                            ), array (
                                                    'escape' =>
false
                                            ) );
                                    }
```

En cambio, el administrador podrá obtener un listado de todas las facturas desde el área administración.



3.4.10 Métodos de pago

Existe un método Pay en el controlador de PaymentData que obtiene el código del método de pago seleccionado (Tpv, Paypal o domiciliación bancaria) actualizando el campo Ticket y el importe para personalizar el proceso de pago.

Se tomo la decisión de almacenar el código php y html de los métodos de pago en la base de datos para poder incluir desde el área de administración nuevos métodos de pago.

El ticket debe ser un número único que identifique inequívocamente la transacción, por eso se ha elegido la fecha completa con el método date('ymdHis').

```
public function pay($method = null, $id = null) {
            $this->PaymentData->PaymentMethod->recursive = 0;


            if (! $this->PaymentData->exists ( $id )) {
                  throw new NotFoundException ( __ ( 'Invalid payment
data id' ) );
            }
            $options = array (
                        'conditions' => array (
```

```
                                    'PaymentData.' . $this-
>PaymentData->primaryKey => $id
                            )
                );
                $paymentData = $this->PaymentData->find ( 'first',
$options );
                if ($paymentData ['PaymentData'] ['status'] == 1) {
                      $this->Session->setFlash ( __ ( 'The following
invoice has been paid.' ), 'default', array (
                                  'class' => 'alert alert-danger'
                      ) );
                } else {


                      $paymentMethods = $this->PaymentData->PaymentMethod-
>find ( 'all', array (
                                  'conditions' => array (
                                              'name' => $method
                                  )
                      ) );
                      $ticket = date ( 'ymdHis' );
                      $paymentData ['PaymentData'] ['payment_method_id'] =
$paymentMethods [0] ['PaymentMethod'] ['id'];
                      $paymentData ['PaymentData'] ['pay_date'] = date (
'y-m-d' );
                      $paymentData ['PaymentData'] ['ticket'] = $ticket;


                      if (! $this->PaymentData->save ( $paymentData )) {
                            $this->Session->setFlash ( __ ( 'The following
invoice could not be update.' ), 'default', array (
                                        'class' => 'alert alert-danger'
                            ) );
                      }
                      $this->set ( 'paymentData', $paymentData );
                      $this->set ( 'ticket', $ticket );
                      $this->set ( 'paymentMethods', $paymentMethods );
```

```
            }

      }
```

El importe y el ticket deben ser actualizados en el código del pago, por lo que se utiliza el método str_replace de php para sustituir los valores en amounttochange y tickettochange. Posteriormente, se realiza una llamada a la función Eval de php para generar el código de salida del método de pago.

```
if (isset ( $paymentData ) && isset ( $paymentMethods )) {

      foreach ( $paymentMethods as $paymentMethod ) {

            $paymentMethod ['PaymentMethod'] ['code'] = str_replace (
"amounttochange", $paymentData ['PaymentData'] ['amount'],
$paymentMethod ['PaymentMethod'] ['code'] );

            $paymentMethod ['PaymentMethod'] ['code'] = str_replace (
"tickettochange", $ticket, $paymentMethod ['PaymentMethod']
['code'] );



            if (isset ( $paymentData ['PaymentData']
['bank_account_number'] )) {

                  $ncc_hidden_begin = substr ( $paymentData
['PaymentData'] ['bank_account_number'], 0, 4 );

                  $ncc_hidden_end = substr ( $paymentData
['PaymentData'] ['bank_account_number'], - 2 );

                  $paymentMethod ['PaymentMethod'] ['code'] =
str_replace ( "ncctochange", $ncc_hidden_begin . " **** **** **" .
$ncc_hidden_end, $paymentMethod ['PaymentMethod'] ['code'] );

            }



            eval ( $paymentMethod ['PaymentMethod'] ['code'] );

            echo $output;

      }

}
```

3.4.10.1 Tpv

El pago por tarjeta puede resultar más complejo de implementar. Necesita un mayor número de variables como se puede observar en la siguiente imagen. Además, se realiza el envío de una firma generada con los datos enviados. Se establece así en el protocolo de comunicación con el banco para garantizar la seguridad de las transacciones.

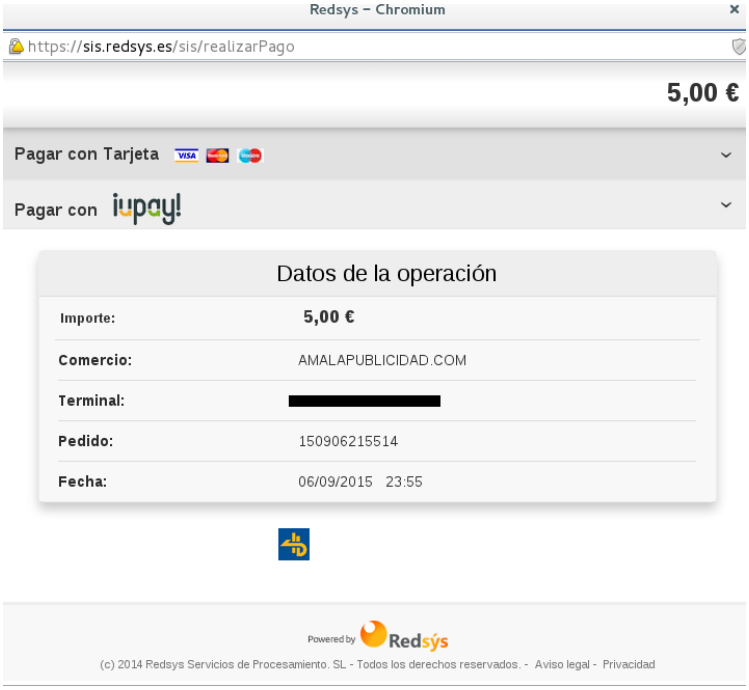
Para poder utilizar este sistema de pago se deberá contactar con un banco para solicitar esta plataforma de pago sujeto a unas condiciones y unas comisiones diferentes en cada entidad bancaria. Redsys[27] se encarga de realizar la comunicación segura entre la aplicación y el banco tanto nuestro como de la persona que realiza el pago.

```
$output = "
            <script language='JavaScript'>
            function calct() {

vent=window.open('','tpv','width=725,height=600,scrollbars=no,resizabl
e=yes,status=yes,menubar=no,location=no');
            document.forms['#formtpv'].submit();
}</script>";
$url_tpvv = 'https://sis.redsys.es/sis/realizarPago';
$clave = '**********';
$name = 'amalapublicidad';
$code = '********';
$terminal = '1';
$order = tickettochange;
$amount = amounttochange * 100;
$currency = '978';
$transactionType = '0';
$urlMerchant = 'http://www.buscopsicologos.es/payment_data/tpv';
$urlOK = 'http://www.buscopsicologos.es/';
$urlKO = 'http://www.buscopsicologos.es/';
$output .= "<form id='formtpv' name=compra action=$url_tpvv
method=post target=tpv> ";
$output .= "
<input type=hidden name=Ds_Merchant_Amount value='$amount'>
<input type=hidden name=Ds_Merchant_Currency value='$currency'>

<input type=hidden name=Ds_Merchant_Order  value='$order'>
<input type=hidden name=Ds_Merchant_MerchantCode value='$code'>
```

27 http://www.redsys.es/

```
<input type=hidden name=Ds_Merchant_Terminal value='$terminal'>

<input type=hidden name=Ds_Merchant_TransactionType
value='$transactionType'>

<input type=hidden name=Ds_Merchant_MerchantURL value='$urlMerchant'>


<input type=hidden name=Ds_Merchant_UrlOK value='$urlOK'>

<input type=hidden name=Ds_Merchant_UrlKO value='$urlKO'>";

$output .= "<input type=hidden name=Ds_Merchant_MerchantData
value='tickettochange' >";

// Compute hash to sign form data

// $signature=sha1_hex($amount,$order,$code,$currency,$clave);

$message = $amount . $order . $code . $currency . $transactionType .
$urlMerchant . $clave;

$signature = strtoupper ( sha1 ( $message ) );

$output .= "<input type=hidden name=Ds_Merchant_MerchantSignature
value='$signature'>";

$output .= '<input type="submit" onclick="calct()" id="continuar"
name="submit" value="Accept" >';

$output .= "</form>";
```

Si todo funciona correctamente, se podrá abrir la ventana de Redsys con el proceso de pago de la transacción.



En el controlador de PaymentData existe un método Tpv que gestiona la respuesta del banco con el resultado de la operación. Si todo ha funcionado correctamente, se almacena el resultado en la transacción.

```
public function tpv() {
            $this->layout = 'clean';
            try {
                  /* por si hay algun fallo se captura las excepciones
*/

                  if (isset ( $_POST ['Ds_Signature'] )) {
                        /* creamos las variables para usar */
                        $Ds_Response = $_POST ['Ds_Response'];
                        // codigo de respuesta
                        $Ds_Amount = $_POST ['Ds_Amount'];
                        // cantidad de la orden
                        $Ds_Order = $_POST ['Ds_Order'];
                        // numero de orden
                        $ticket = $Ds_Order;
                        $Ds_MerchantCode = $_POST ['Ds_MerchantCode'];
                        // codigo de comercio
                        $Ds_Currency = $_POST ['Ds_Currency'];
                        // moneda
                        $firmaBanco = $_POST ['Ds_Signature'];
                        // firma hecha por el banco
                        $Ds_Date = $_POST ['Ds_Date'];
                        // fecha
                        $CLAVE = '**********';
                        // clave secreta proporcionada por el banco

                        $MerchantData = $_POST ['Ds_MerchantData'];

                        /* creamos la firma para comparar */
```

```
                        $firma = strtoupper ( sha1 ( $Ds_Amount .
$Ds_Order . $Ds_MerchantCode . $Ds_Currency . $Ds_Response .
$CLAVE ) );


                        $Ds_Response += 0; // convertimos la respuenta
en un numero concreto.


                        if ($firma == $firmaBanco && $Ds_Response >= 0
&& $Ds_Response <= 99) // && $Ds_Response>=0 && $Ds_Response<=99
{
                                // LA RESPUESTA DEL BANCO ES AUTÉNTICA,
GESTIONAMOS EL PEDIDO. DS_Response de 0 a 99 indica que la transaccion
ha sido realizada con exito


                                $ticket = $Ds_Order;
                                $options = array (
                                            'conditions' => array (
                                                        'PaymentData.tic
ket' => $ticket
                                            )
                                );
                                $paymentData = $this->PaymentData->find
( 'first', $options );


                                $paymentData ['PaymentData'] ['status']
= 1;


                                if (! $this->PaymentData->save
( $paymentData )) {
                                }
                        }
                }
            }


            catch ( Exception $e ) {
```

```
            }
      }
```

3.4.10.2 Paypal

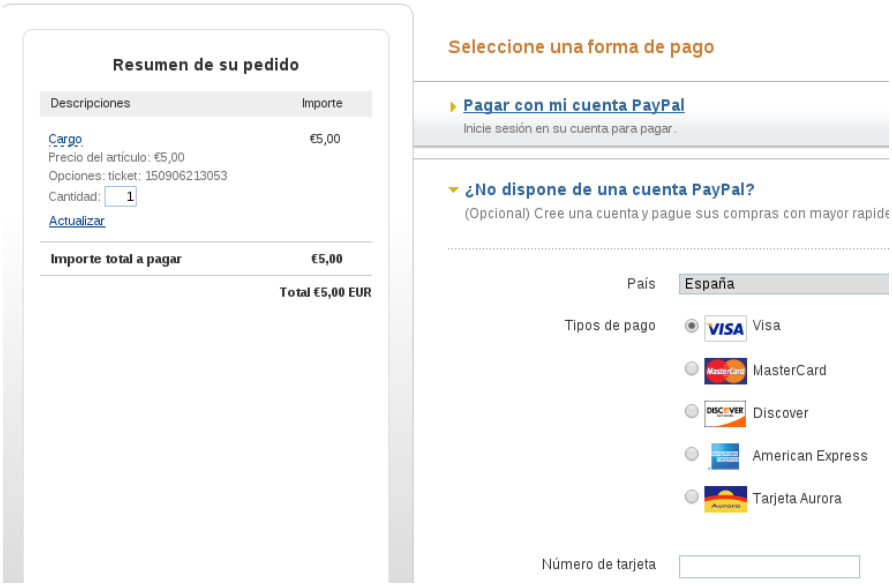
Para recibir un pago a través de Paypal simplemente se deberá abrir una cuenta de Paypal de tipo comercio para recibir ingresos. Existen unas tarifas por el cobro a través de esta plataforma y no puede utilizarse para vender ciertos productos que pueden incumplir sus condiciones.

En la siguiente imagen, se adjunta el código para crear un pago mediante Paypal.

```
$output = "<script language='JavaScript'>
            function calcp() {
vent=window.open('','paypal','width=825,height=700,scrollbars=no,resiz
able=yes,status=yes,menubar=no,location=no');
            document.forms['#formpaypal'].submit();
            }
            </script>";

      $output .= '
            <form id="formpaypal" action="https://www.paypal.com/cgi-
bin/webscr" method="post" target=paypal>
            <input type="hidden" name="cmd" value="_xclick">
            <input type="hidden" name="business"
value="info@**********.com" >
            <input type="hidden" name="item_name" value="Cargo">
            <input type="hidden" name="currency_code" value="EUR">
            <input type="hidden" name="undefined_quantity" value="1">
            <input type="hidden" name="amount" value="amounttochange">
            <input type="hidden" name="on0" value="ticket">
            <input type="hidden" name="os0" value="tickettochange" >
            <input type="hidden" name="return"
value="http://www.buscopsicologos.es/"  />
            <input type="hidden" name="cancel_return"
value="http://www.buscopsicologos.es/"  />
```

```
            <input type="hidden" name="notify_url"
value="http://www.buscopsicologos.es/payment_data/ipn"  />

            <input type="submit" onclick="calcp()" id="continuar"
name="submit" value="Accept">

            </form>';
```

Obteniendo satisfactoriamente una ventana de pago por paypal



Finalmente, en la imagen podemos observar el código para la recepción del comunicado por paypal, donde se especifica si un pago ha sido completado con éxito y donde la aplicación archiva el pago como pagado.

```
public function ipn() {
            $this->layout = 'clean';
            // read the post from PayPal system and add 'cmd'
            $req = 'cmd=_notify-validate';

            foreach ( $_POST as $key => $value ) {
                  $value = urlencode ( stripslashes ( $value ) );
                  $req .= "&$key=$value";
```

```
            }


            // post back to PayPal system to validate
            $header = "POST /cgi-bin/webscr HTTP/1.0\r\n";
            $header .= "Content-Type: application/x-www-form-
urlencoded\r\n";
            $header .= "Content-Length: " . strlen ( $req ) .
"\r\n\r\n";
            $fp = fsockopen ( 'ssl://www.paypal.com', 443, $errno,
$errstr, 30 );


            if (! $fp) {
                  // HTTP ERROR
            } else {
                  fputs ( $fp, $header . $req );
                  while ( ! feof ( $fp ) ) {
                        $res = fgets ( $fp, 1024 );
                        if (strcmp ( $res, "VERIFIED" ) == 0) {


                              // check the payment_status is Completed


                              // check that txn_id has not been
previously processed


                              // check that receiver_email is your
Primary PayPal email


                              // check that
payment_amount/payment_currency are correct
                              // process payment


                              if (isset ( $_POST ['payment_date'] ) &&
isset ( $_POST ['txn_id'] ) && isset ( $_POST ['payment_status'] )) {
```

```
                                    // assign posted variables to
local variables

                                    $item_name = $_POST ['item_name'];

                                    $item_number = $_POST
['item_number'];

                                    $payment_status = $_POST
['payment_status'];

                                    $payment_amount = $_POST
['mc_gross'];

                                    $payment_currency = $_POST
['mc_currency'];

                                    $txn_id = $_POST ['txn_id'];

                                    $receiver_email = $_POST
['receiver_email'];

                                    $payer_email = $_POST
['payer_email'];


                                    $payment_date = $_POST
['payment_date'];

                                    $txn_id = $_POST ['txn_id'];

                                    $payment_status = $_POST
['payment_status'];


                                    if ($payment_status = "Completed")
{

                                          $ticket = $_POST
['option_selection1'];


                                          $options = array (

                                                'conditions' =>
array (

                                                      'Pay
mentData.ticket' => $ticket

                                                )

                                          );

                                          $paymentData = $this-
>PaymentData->find ( 'first', $options );
```

```
                                        $paymentData ['PaymentData']
['status'] = 1;
                                        if (! $this->PaymentData-
>save ( $paymentData )) {
                                        }
                                    }
                                }
                            } else if (strcmp ( $res, "INVALID" ) == 0) {
                                // log for manual investigation
                            }
                        }
                        fclose ( $fp );
                    }
                }
```

3.4.10.3 Domiciliación bancaria

Existe un método sepa en el controlador de PaymentData que simplemente almacena en la base de datos el número de cuenta donde el usuario desea domiciliar el recibo. De este modo, la empresa se encargaría de ordenar el cobro de los recibos mensualmente en la cuenta indicada.

```
public function sepa() {
            if ($this->request->is ( array (
                        'post',
                        'put'
            ) )) {
                  $ticket = $this->request->data ['ticket'];
                  $ncc = $this->request->data ['ncc'];

                  $options = array (
```

```
                                'conditions' => array (
                                                'PaymentData.ticket' =>
$ticket
                                )
                );
                $paymentData = $this->PaymentData->find ( 'first',
$options );

                $paymentData ['PaymentData'] ['ticket'] = $ticket;
                $paymentData ['PaymentData'] ['bank_account_number']
= $ncc;

                if (! $this->PaymentData->save ( $paymentData )) {
                        $this->Session->setFlash ( __ ( 'The following
invoice could not be update.' ), 'default', array (
                                        'class' => 'alert alert-danger'
                        ) );
                }
        }
    }
```

3.4.11 Otras modificaciones en el código

3.4.11.1 Función create del controlador de facturas (Bill).

En el controlador de facturas existe una función create para generar el listado de facturas de forma periódica. Se puede ejecutar la orden de forma manual o programar la ejecución con CRON[28].

La creación automática de facturas es importante para simplificar el proceso y no ser obligados a generar manualmente las nuevas entradas del modelo Bills.

28 https://es.wikipedia.org/wiki/Cron_(Unix)

```
public function create() {

            $this->Bill->recursive = 4;

            $contracts = $this->Bill->Contract->find ( 'all' ); //
buscamos todos los contratos

            foreach ( $contracts as $contract ) {

                  if ($contract ['Status'] ['name'] == "Activa") { //
sólo lo aplicamos a los contratos activos

                        $amount = $contract ['ContractType']
['price']; // mensualidad del contrato



                        $this->Bill->PaymentData->create (); //
creamos una nueva linea de pago PaymentData

                        if (isset ( $contract ['Contract']
['bank_account_number'] ) && ! empty ( $contract ['Contract']
['bank_account_number'] )) { // si tenemos almacenado el número de
cuenta, será porque desea domiciliar el recibo

                              $this->Bill->PaymentData->set (
'bank_account_number', $contract ['Contract']
['bank_account_number'] );

                              $paymentMethods = $this->Bill-
>PaymentData->PaymentMethod->find ( 'all', array (

                                    'conditions' => array (

                                          'name' => 'Sepa'

                                    )

                              ) );

                        } else {

                              $paymentMethods = $this->Bill-
>PaymentData->PaymentMethod->find ( 'all', array (

                                    'conditions' => array (

                                          'name' => 'Tpv'

                                    )

                              ) );

                        }



                        $ticket = date ( 'ymdHis' ); // generamos un
nuevo ticket para identificar el proceso de pago
```

```
                                                    // Preparamos los
campos del modelo para ser almacenado en la base de datos
                        $this->Bill->PaymentData->set ( 'amount',
$amount );
                        $this->Bill->PaymentData->set (
'payment_method_id', $paymentMethods [0] ['PaymentMethod'] ['id'] );
                        $this->Bill->PaymentData->set ( 'pay_date',
date ( 'y-m-d' ) );
                        $this->Bill->PaymentData->set ( 'ticket',
$ticket );
                        $this->Bill->PaymentData->set ( 'status', 0 );


                        if ($this->Bill->PaymentData->save ()) { // si
ha sido guardado, consultamos el id para realizar las relaciones
                              $lastid = $this->Bill->PaymentData-
>getLastInsertID ();


                              // Debemos preparar el campo Record,
consultando la última entrada
                              $lastbill = $this->Bill->find ( 'first',
array (
                                          'conditions' => array (
                                                'Bill.record
LIKE' => '%/A/' . date ( 'y' ) . '%'
                                          ),
                                          'order' => array (
                                                'Bill.id' =>
'DESC'
                                          )
                              ) );


                              if (empty ( $lastbill )) {
                                    $currentrecord = 1; // si no hay
facturas en un año, empezamos con 1
                              } else { // si las hay, sumamos 1 a la
anterior factura
```

```
                                    $lastbillrecord = explode ( "/",
$lastbill ['Bill'] ['record'] );

                                    $currentrecord = ($lastbillrecord
[0] + 1) . "/A/" . date ( 'y' ); // Por ejemplo: 210/A/15

                                    }


                                    $this->Bill->create (); // Preparamos la
factura

                                    $this->Bill->set ( 'payment_data_id',
$lastid );

                                    $this->Bill->set ( 'bill_date', date (
'y-m-d' ) );

                                    $this->Bill->set ( 'contract_id',
$contract ['Contract'] ['id'] );

                                    $this->Bill->set ( 'record',
$currentrecord );


                                    if ($this->Bill->save ()) {

                                          $lastid = $this->Bill-
>getLastInsertID ();

                                          // Insertamos en BillsContract la
relación entre Factura y Contrato

                                          $this->Bill->BillsContract->create
();

                                          $this->Bill->BillsContract->set (
'contract_id', $contract ['Contract'] ['id'] );

                                          $this->Bill->BillsContract->set (
'bill_id', $lastid );

                                          if ($this->Bill->BillsContract-
>save ()) {


                                                $this->flash ( __ ( 'The
bill has been created.' ), array (

                                                            'action' =>
'index'

                                                ) );

                                          }

                                    }
```

```
                        }
                    }
                }
            }
```

Se genera de forma automática lo necesario para crear las facturas de cada contrato que este activo. Se completa los campos de Bill y de PaymentData, creando las relaciones necesarias siguiendo el modelo de la base de datos.

Se utiliza el método getLastInsertID para obtener el último id insertado en un modelo, y así crear la relación con otra entidad.

3.4.11.2 Función value del controlador de puntuaciones (Rank).

Con esta función se añade la puntuación recibida por un visitante de la web sobre un psicólogo. Se realiza la suma de todas las puntuaciones y se contabiliza el número total para realizar la media del psicólogo.

```
public function value($id = null, $rank = null) {
            $this->layout = 'json';


            if (($rank > 0 && $rank < 6)) {
                  $data ['User'] ['id'] = $id;
                  $data ['Rank'] ['score'] = $rank;


                  $options = array (
                              'conditions' => array (
                                          'User.id' => $id
                              )
                  );
                  $storedranks = $this->Rank->find ( 'all',
$options ); // consultamos todas las puntuaciones almacenadas
                  $cont = 1; // empezamos con 1 para contar la
puntuacion actual y guardarla junto el resto de las puntuaciones
                  $sum = $rank;
                  foreach ( $storedranks as $storedrank ) {
```

```
                        $sum += $storedrank ['Rank'] ['score'];
                        $cont ++;
                    }

                    $data ['User'] ['score_average'] = ceil ( $sum /
$cont );

                    if (! $this->Rank->saveAssociated ( $data )) {
                        $this->Session->setFlash ( __ ( 'The rank
could not be saved. Please, try again.' ), 'default', array (
                                'class' => 'alert alert-danger'
                        ) );
                    } else {
                        $this->set ( 'score_average', $data ['User']
['score_average'] );
                        $this->set ( 'id', $data ['User'] ['id'] );
                    }
                }
            }
```

3.4.11.3 Función contact del controlador User.

Este método se utiliza como medio de contacto entre el visitante y el psicólogo, y se debe destacar porque se utiliza el componente CakeEmail para realizar el envío del mensaje por correo electrónico.

```
public function contact($id = null) {
            $this->layout = 'ajax';

            if (! $this->User->exists ( $id )) {
                throw new NotFoundException ( __ ( 'Invalid user'
) );
            }
            $options = array (
                    'conditions' => array (
```

```
                                    'User.' . $this->User->primaryKey
=> $id
                        )
            );
            $user = $this->User->find ( 'first', $options );
            $this->set ( 'user', $user );

            if ($this->request->is ( array (
                        'post',
                        'put'
            ) )) {

                  $this->request->data ['User'] ['id'] = $id;
                  if (! $this->User->Question->saveAssociated ( $this-
>request->data )) {
                        $this->Session->setFlash ( __ ( 'The visitor
could not be saved. Please, try again.' ), 'default', array (
                                    'class' => 'alert alert-danger'
                        ) );
                  } else {
                        // mensaje para el usuario
                        $Email = new CakeEmail ( 'smtp' );
                        $Email->to ( $user ['User'] ['email'] );
                        $Email->subject ( 'Contact message from ' .
$this->data ['Visitor'] ['firstname'] . ' ' . $this->data ['Visitor']
['lastname'] );
                        $Email->from ( 'info@buscopsicologos.es' );
                        $message = "Hola " . $user ['User']
['firstname'] . ",\r\n\r\n";
                        $message .= "Has recibido un mensaje de
contacto desde www.buscopsicologos.es\r\n\r\n";
                        $message .= "El contenido del mensaje ha
sido:\r\n";
                        $message .= $this->data ['Question']
['message'];
                        $Email->send ( $message );
```

```
                        // mensaje copia para el visitante
                        $Email = new CakeEmail ( 'smtp' );
                        $Email->to ( $this->data ['Visitor'] ['email']
);
                        $Email->subject ( 'Contact message from ' .
$this->data ['Visitor'] ['firstname'] . ' ' . $this->data ['Visitor']
['lastname'] );
                        $Email->from ( 'info@buscopsicologos.es' );
                        $message = "Hola " . $this->data ['Visitor']
['firstname'] . ",\r\n\r\n";
                        $message .= "Has enviado un mensaje de
contacto desde www.buscopsicologos.es\r\n\r\n";
                        $message .= "El contenido del mensaje ha
sido:\r\n";
                        $message .= $this->data ['Question']
['message'];
                        $Email->send ( $message );
                }
            }
      }
```

Se almacena el mensaje en la base de datos para que los administradores puedan tener un registro de los movimientos de la web, y se envía por correo una copia del mensaje al usuario.

Para que el envío sea posible, será necesario especificar en el fichero app/Config/email.php la configuración de una cuenta de correo y un servidor smtp.

```
public $smtp = array (
                'transport' => 'Smtp',
                'from' => array (
                            'info@buscopsicologos.es' =>
'Buscopsicólogos'
                ),
                'host' => '**************',
                'port' => 25,
```

```
                'timeout' => 30,
                'username' => 'info@buscopsicologos.es',
                'password' => '**********',
                'client' => null,
                'log' => false
    )
    ;
```

3.4.12 Creación de gráficos vectoriales

También se ha realizado el diseño de un logotipo para la página web utilizando el software libre Inkscape representando el nombre que se asignará al proyecto: "Buscopsicólogos".

Esta imagen ha sido creada utilizando la letra ψ "psi" del alfabeto griego. Carácter utilizado como símbolo de la psicología y de la psiquiatría.

Y la imagen de la lupa[29] ha sido obtenida de la Wikimedia Commons, donde su uso está disponible para su modificación incluso con fines comerciales.

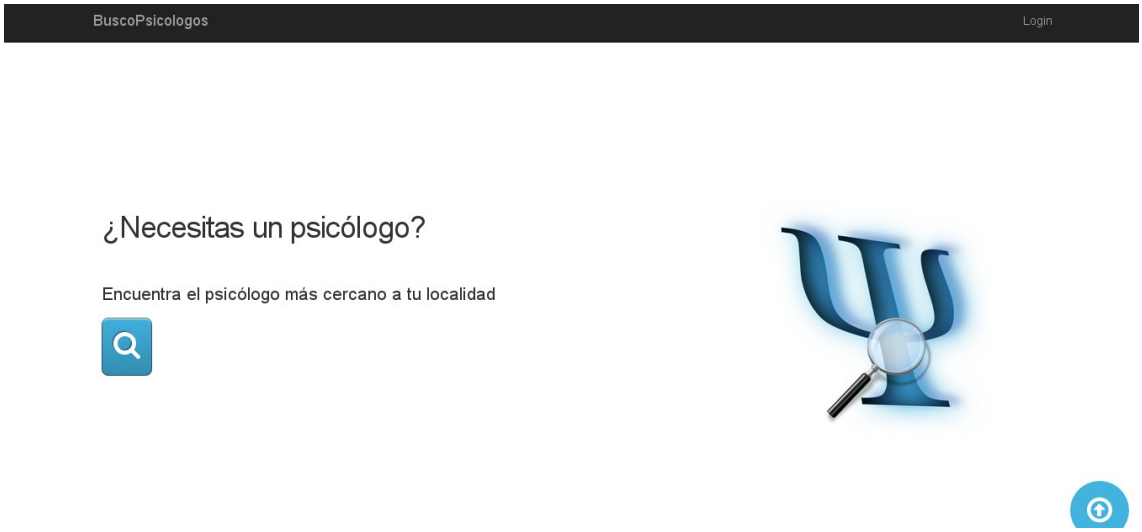
29 By AlphaZeta (Own work) [CC0], via Wikimedia Commons

4 Test y pruebas

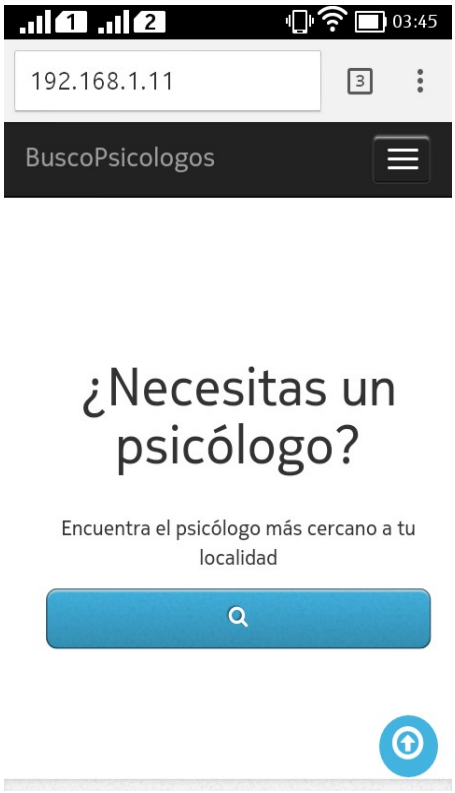
A continuación, se adjuntaran imágenes con algunos de los procesos desarrollados en la aplicación durante su funcionamiento.

4.1 Visualización de la web en diferentes dispositivos

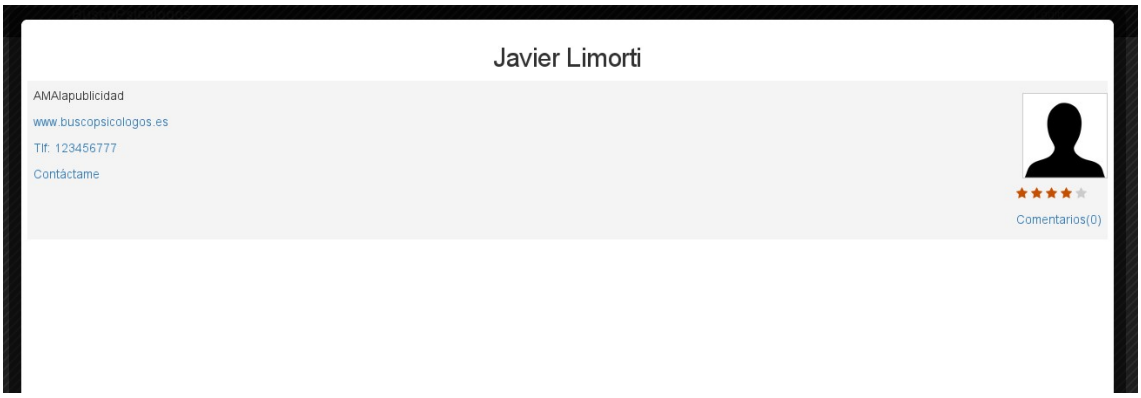
4.1.1 Ordenador portátil de 15.4 pulgadas



4.1.2 Dispositivo móvil Android de 4.3 pulgadas.



4.2 Búsqueda de un psicólogo en Valencia, contacto y valoración del profesional



4.3 Visualizar una factura en pdf con la plantilla de la empresa.

Se han eliminado algunas zonas de la factura para respetar los datos fiscales de la empresa.

FACTURA

Factura nº: 2/A/15
Fecha: 2015-08-27

AMAlapublicidad

[redacted]
Dirección: [redacted]
Población: Valencia
Provincia: Valencia
Código Postal: [redacted]
Teléfono: [redacted]
N.I.F./C.I.F.: [redacted]

Datos del cliente

Nombre: Javier
Apellidos: Limorti
Dirección: Plaza del Ayuntamiento nº 19 10ºA
Población: Valencia del Mombuey
Provincia: Badajoz
Código Postal: 6134
N.I.F./C.I.F.: 298Q

CONCEPTO		IMPORTE (€)
Servicios de www.buscopsicologos.es		19.01 €
	Base imponible:	19.01 €
	I.V.A: 21%	3.99 €
	TOTAL:	23 €

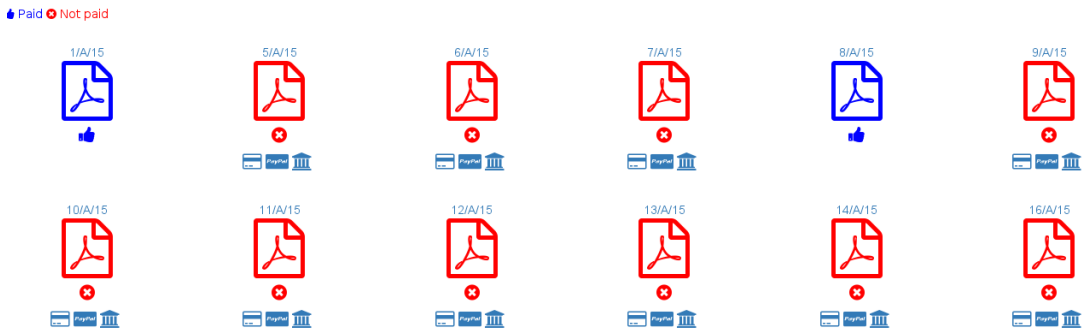
[redacted]

El recibo bancario es el justificante de pago de esta factura.

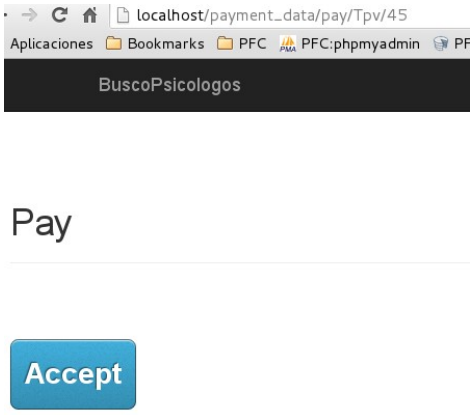
De acuerdo con la Ley Orgánica de Protección de Datos 15/1999, le comunicamos que sus datos forman parte de los ficheros automatizados y no automatizados Podrá ejercer sus derechos de acceso, cancelación, rectificación y oposición escribiéndonos a Agencia de Marketing de Autónomos (AMA), Plaza del Ayuntamiento, número 19, 10ºA , 46002 Valencia, o bien por e-mail: info@amalapublicidad.com

4.4 Pago de una factura mediante tarjeta de crédito/débito

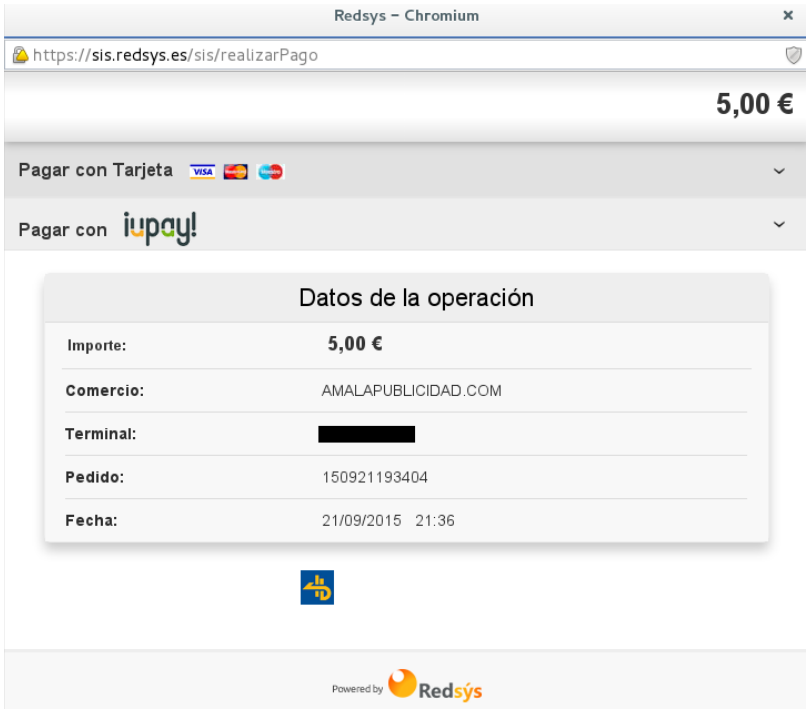
En el listado de facturas, las de color rojo son las que no han sido abonadas aun. Se puede pulsar en la opción de pago por tarjeta.



Se mostrará un boton “Accept” para aceptar el pago.



Se abrirá una ventana en el navegador con los datos del pago. La URL de esta ventana procede de redsys para garantizar la seguridad del pago.



Por defecto, se ofrece la opción de pago por iupay[30] como método de pago. Esta plataforma fue lanzada recientemente para aumentar la seguridad de los pagos con tarjeta al registrarse con usuario y contraseña sin la necesidad de introducir los datos bancarios en cada pago.

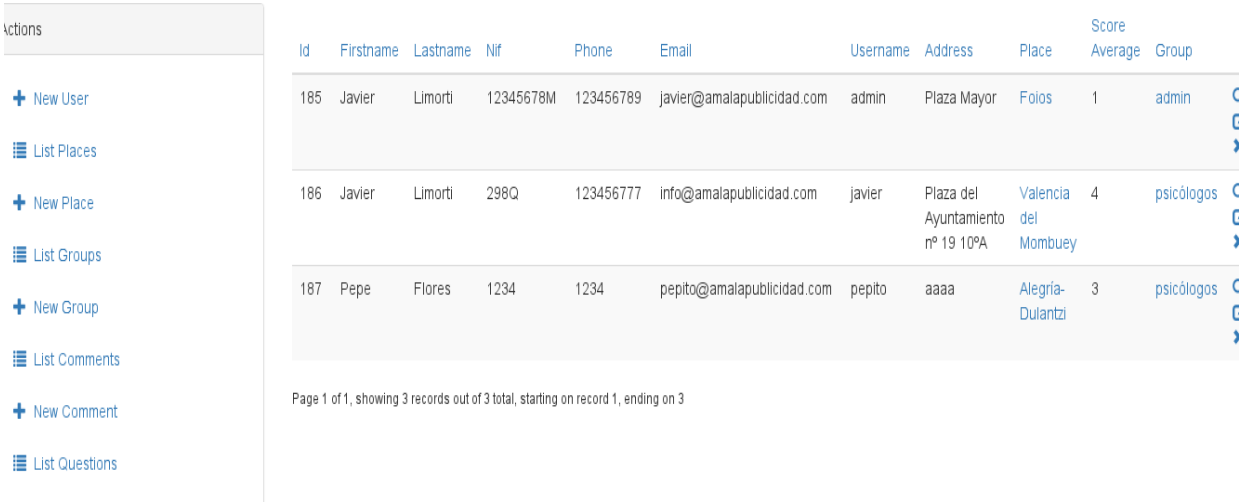
Al ser un método relativamente reciente, la empresa suele sugerir a sus clientes el uso del método tradicional de pago con tarjeta. Donde se introducen los datos de la tarjeta de crédito/débito, la fecha de caducidad y el CVV[31] de la tarjeta.

Al completar la operación con los datos de la tarjeta, los sistemas de redsys enviarán un mensaje de notificación a la url especificada. Donde la aplicación actualizará los datos de la transacción marcando el estado como pagado.

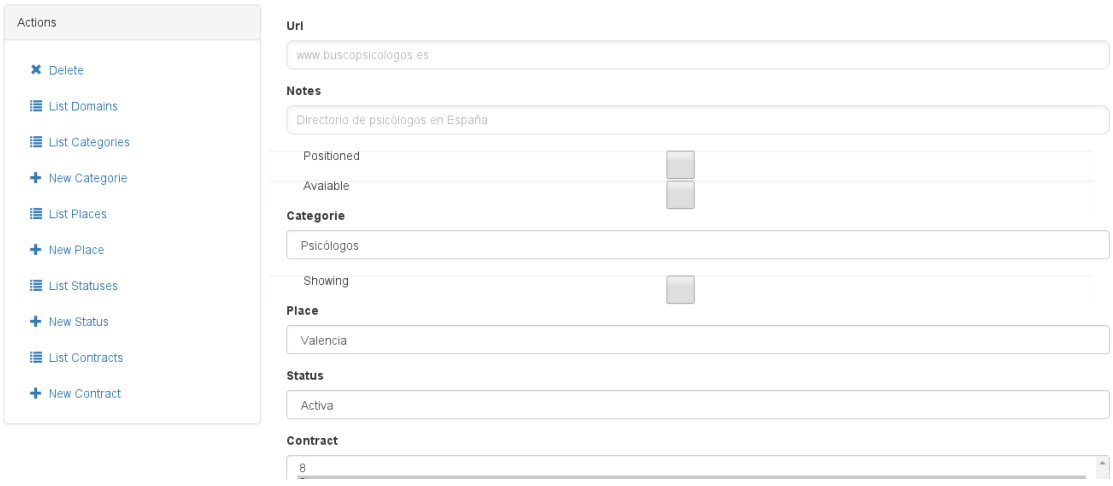
30 https://www.iupay.es

31 https://en.wikipedia.org/wiki/Card_security_code

4.5 Listar los clientes de la empresa



4.6 Editar la información de un dominio



5 Conclusiones

Me gustaría destacar del presente Proyecto Final de Carrera la experiencia positiva que ha resultado la utilización del framework CakePHP.

Durante el desarrollo de la aplicación me he encontrado con obstáculos que gracias al esfuerzo y al tiempo dedicado he conseguido superar satisfactoriamente.

Gracias a ello, he conseguido mejorar mi modo de desarrollar los proyectos y cómo abstraer de los requisitos todo lo necesario para la implementación del código cumpliendo las necesidades de la empresa y cómo anticiparme a posibles cambios en los requisitos generalizando cada requisito para facilitar la evolución del software.

CakePHP ofrece multitud de posibilidades para realizar cualquier proyecto informático y facilita en gran medida la posible evolución del software en un futuro.

Me gustaría añadir que en un futuro la aplicación será mejorada con nuevas funcionalidades que actualmente no se han incluido por no ser necesarias en la empresa.

Por ejemplo, se permitirá a los psicólogos editar su propia información del perfil. Además, tendrán la posibilidad de gestionar el contenido de su página web a través del área de administración del directorio de psicólogos. Y el administrador dispondrá de una plataforma para enviar mensajes personalizados a sus clientes desde la aplicación seleccionando un grupo de destinatarios de la base de datos.

Con la aplicación, la empresa se beneficiará de la aplicación al disponer de una cartera de clientes que puedan aparecer en la web desde el primer momento.

Lo realmente interesante del presente Proyecto Final de Carrera es la posibilidad de multiplicar los beneficios de la empresa creando nuevos directorios de profesionales de otros ámbitos como fontaneros, electricistas, etc.

Por lo tanto, puedo afirmar que me siento satisfecho al haber realizado la aplicación y que me ha aportado una gran experiencia durante su desarrollo para desenvolverme en el mundo laboral con mayor habilidad.

6 Bibliografía y enlaces

- https://github.com/sams/alaxos_acl
- https://github.com/josegonzalez/cakephp-upload
- http://api.jquery.com/
- http://www.jacklmoore.com/colorbox/
- http://datatables.net/
- https://fortawesome.github.io/Font-Awesome/
- http://html5up.net/
- https://es.wikipedia.org
- https://canales.redsys.es/
- https://developer.paypal.com
- http://www.fpdf.org/
- http://php.net/manual/es/index.php
- CakePHP Cookbook Documentation Release 2.x
    - http://book.cakephp.org/2.0/_downloads/en/CakePHPCookbook.pdf
- Una guía para la realización y supervisión de proyectos final de carrera (PFC) en el ámbito de la web por “Félix Buendía Garcia”.

7 Glosario

1 Paypal [https://www.paypal.com]

2 Framework [https://es.wikipedia.org/wiki/Framework]

3 Cakephp [http://cakephp.org/]

4 PHP [https://es.wikipedia.org/wiki/PHP]

5 URL [https://es.wikipedia.org/wiki/Localizador_de_recursos_uniforme]

6 Reponsive [http://es.wikipedia.org/wiki/Diseño_web_adaptable]

7 MySQL Workbench [https://www.mysql.com/products/workbench/]

8 CakePHP conventions [http://book.cakephp.org/2.0/es/getting-started/cakephp-conventions.html]

9 Apache [https://es.wikipedia.org/wiki/Servidor_HTTP_Apache]

10 MySQL [https://es.wikipedia.org/wiki/MySQL]

11 PhpMyAdmin [https://es.wikipedia.org/wiki/PhpMyAdmin]

12 Debian [https://www.debian.org]

13 Hash [https://es.wikipedia.org/wiki/Función_hash]

14 Mod Rewrite [http://httpd.apache.org/docs/current/mod/mod_rewrite.html]

15 jQuery [https://jquery.com/]

16 HTML [https://es.wikipedia.org/wiki/HTML]

17 Bootstrap [http://getbootstrap.com/]

18 Font Awesome [https://fortawesome.github.io/Font-Awesome/]

19 Datatables [https://www.datatables.net/]

20 jQuery Ajax [http://api.jquery.com/jquery.ajax/]

21 JSON [https://es.wikipedia.org/wiki/JSON]

22 Colorbox [http://www.jacklmoore.com/colorbox/]

23 Plugin ACL alaxos [https://github.com/sams/alaxos_acl]

24 Plugin CakePHP-upload [https://github.com/josegonzalez/cakephp-upload]

25 Documentación CakePHP-upload [http://cakephp-upload.readthedocs.org/en/latest/index.html]

26 Plugin FPDF [http://www.fpdf.org/]

27 Redsys [http://www.redsys.es/]

28 CRON [https://es.wikipedia.org/wiki/Cron_(Unix)]

29 Lupa [https://commons.wikimedia.org/wiki/File:Magnifying_glass_icon_mgx2.svg]

30 IuPay [https://www.iupay.es]

31 CVV - https://en.wikipedia.org/wiki/Card_security_code