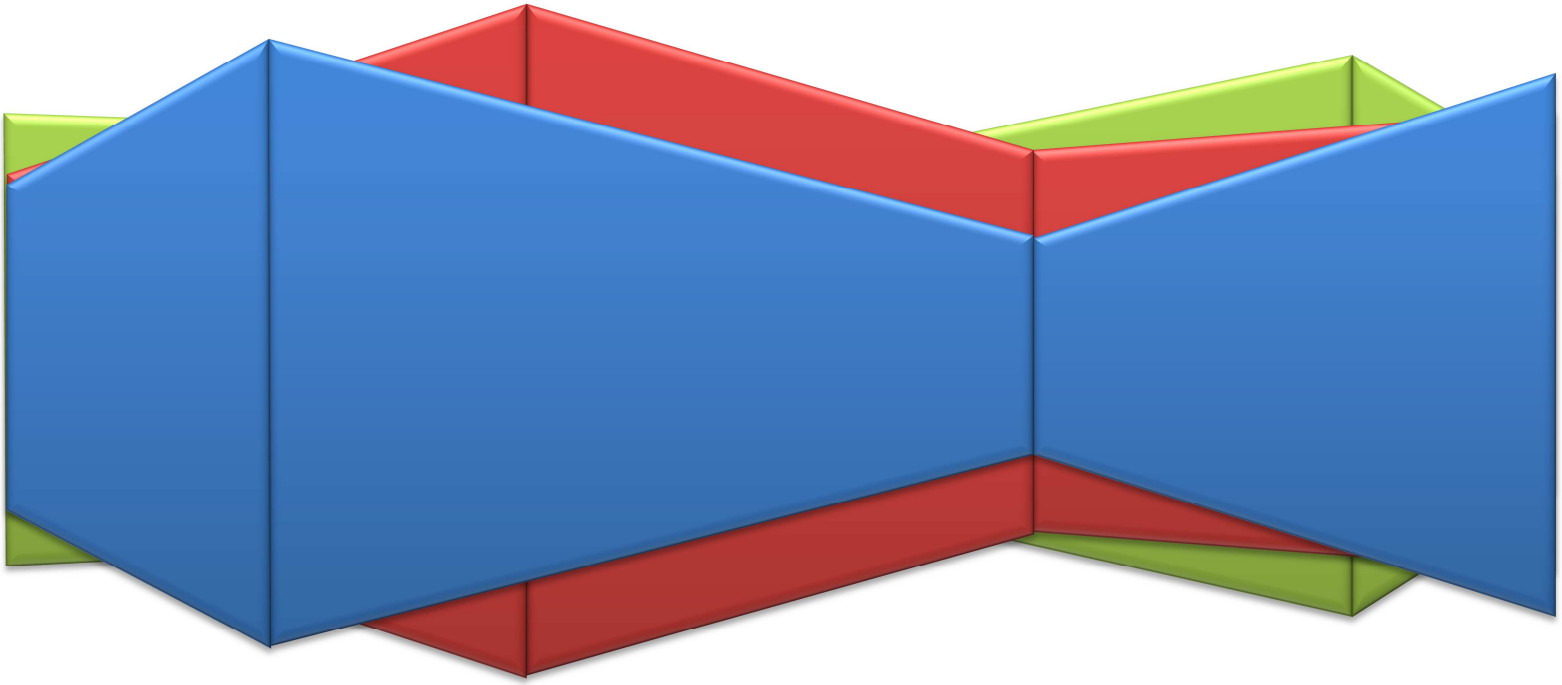
Desarrollo de Aplicaciones Multiplataforma

Dame la patita

Documentación del proyecto final

José Ignacio Claver Paules



Índice

Objetivos

Dame la patita tiene como su principal objetivo crear un espacio donde se puedan anunciar en exclusiva tres tipos de publicaciones:

- ➔ Animales en adopción
- ➔ Animales que se han perdido
- ➔ Animales que se han encontrado en la calle.

La idea básica es crear un portal de anuncios centrado en estos tres tipos de publicaciones, facilitando la búsqueda a cualquier usuario que desee un servicio similar.

El alcance actualmente es a nivel de España. La idea está aún en pañales y requiere de un crecimiento lento pero seguro. La adopción de animales está muy extendida pero tremendamente esparcida a lo largo de muchas redes sociales y páginas web.

Ahí es donde entra el segundo objetivo de mi proyecto. Centralizar toda la información en un mismo lugar que, primero, facilite la vida a los usuarios y segundo, evite la duplicidad de anuncios y ser constantemente bombardeado en otras páginas con publicaciones que no pueden interesarte y llevan a ignorarlas.

A largo plazo los objetivos son dos:

- ➔ Expandirse en el uso de funcionalidades de geoposicionamiento y avisos, permitiendo crear alertas a las veterinarias cercanas en caso de que se pierda tu animal.
- ➔ Creación de una red social paralela a la aplicación que permita la comunicación entre usuarios directamente, para compartir información sobre sus mascotas, publicaciones interesantes, creaciones de eventos… Es una idea en pañales y para implementar cuando se alcance una cantidad de usuarios respetable y en un espacio de tiempo considerablemente mayor del otorgado para este proyecto.

La motivación del proyecto es, como la mayoría de las vez que me embarco en un proyecto, la experiencia propia, o en este caso la de una buena amiga.

Se da la situación de que hay ciertas organizaciones de protección animal que se dedican a hacer negocio con aquellos que rescatan, a base de demandas, exigencias de dinero para permitir la adopción y que realmente solo sirven para ejercer de tapón de aquellos que sí quieren ayudar. Y no tenemos que irnos muy lejos, en Huesca hay una muy conocida que lleva a cabo estas prácticas.

Nunca me había planteado como actuar ante el tema hasta que una amiga, por el simple hecho de adoptar un gato, se vio envuelta en demandas y juicios injustos.

Yo poco podía hacer, pero sí está en mi mano poner una pequeña solución para evitar que en el futuro otras personas se vean en la misma situación.

Y de ahí nació la primera idea de ***Dame la patita***. Crear una plataforma para adoptar animales en las que, bajo ningún concepto, se permite el intercambio de dinero. Un sitio donde poder denunciar los anuncios ficticios sin necesidad de represalias y donde la gente pueda buscar hogares para estos seres.

Conclusiones

Ventajas comparativas

SI usas Facebook, Twitter o alguna red social similar lo más probable es que hayas visto muchos anuncios que anuncian animales perdidos o en adopción, ¿verdad? Y también es muy probable que, un mes después de ver el primero, si te percataste de ello, hayas vuelto a ver exactamente la misma publicación, compartida por otra persona totalmente distinta.

Es la maravilla y maldición de estas redes sociales, la información se expande tan rápido que es imposible de controlar. ¿Y qué sucede cuando el animal por fin ha sido adoptado, encontrado…? ¿Cómo se para esa cadena?

Hay es donde entra ***Dame la patita***. Se trata de un sitio donde centralizar todos estos tipos de publicaciones y que baste indicar en tu perfil que dicho anuncio ya no está disponible para finalizar su visualización y dejar de estar accesible para todo el mundo.

Respecto a productos similares, he encontrado varias webs que se centran en la compartición de animales en adopción, como http://www.sanadoptin.org/ o en casos de mascotas perdidas, por ejemplo http://animales-perdidos.org/, pero en general, en todos los sitios que busque, encontré las mismas carencias:

1. Se centran en una sección demasiado exclusiva. O bien solo muestran los animales en adopción de una perrera en exclusiva, o solo en animales perdidos, por ejemplo. No hay ninguna (en España al menos) que agrupe todas las categorías y de una manera sencilla y rápida de ver. Y al mismo tiempo enrevesan toda la aplicación consiguiendo que el usuario acabe desperdigado por la web sin centrarse en la información que quiere transmitir.
2. Otro punto negativo es que muy pocas de estas aplicaciones son *responsive*. Cuando has perdido un animal, o te encontraste con alguno, lo primordial es tener un acceso rápido para denunciar el caso cuanto antes. Se debe facilitar el acceso a la web desde dispositivos móviles, y hacer el acceso a las publicaciones rápido, sencillo e intuitivo.

Contando con estos puntos, ***Dame la patita*** entra en el mercado para cubrir esas carencias (¡o intentarlo!).

Está creada con tecnología *mobile-first*, para pensar en los dispositivos móviles y luego dar el salto a pantallas grandes, hace que publicar un anuncio requiera de tres pasos contados. Registro, login y publicación. Se han realizado pruebas con gente que no había tocado ni visto previamente esta web y en dos minutos el más lento consiguió tener un anuncio publicado, y aun le dio tiempo de curiosear por la aplicación.

Y por supuesto, como comenté en la sección de **OBJETIVOS**, la carencia de cubrir todo tipo de anuncios principales queda cubierta con la idea principal, mostrando una información clara, concisa y muy fácil de interpretar, con una interfaz bastante intuitiva.

Eso considero que le da una ventaja a mi aplicación, pero por supuesto solo podría llevarse a la práctica realizando una campaña publicitaria atractiva para atraer al mayor número de gente. Muchas veces no triunfa quien mejor tecnología tiene sino quien mejor la sabe promocionar.

Dificultades

Realmente, la dificultad ha sido el aprendizaje de como desarrollar toda la página web en sí, principalmente por desconocimiento de las tecnologías web en su mayor medida.

Tengo conocimientos de HTML, CSS y PHP pero los considero básicos, lo que me ha llevado a gastar una ingente cantidad de tiempo en documentarme y profundizar para aprender sobre tecnologías web, y sobre como estructurar un proyecto adecuadamente para realizar un mejor mantenimiento del mismo.

Lo desarrollo más profundamente en secciones posteriores, pero he tenido que aprender sobre Javascript, tecnología AJAX, PHP7, HTML5 y CSS3.

Una vez bien documentado, para facilitar el desarrollo, decidí utilizar Bootstrap para el diseño, con lo que de nuevo fue aprender de cero a emplear un framework que nunca había manejado.

La comunicación con la API de Geoposicionamiento de Google Maps no resultó tan dificultosa como había planteado, pero aún así me genero varios quebraderos de cabeza por no conseguir comunicarla con Javascript directamente.

Y en la parte de elaboración, lo más dificultoso sin duda fue configurar correctamente el servidor Mercury para poder realizar envios desde una cuenta “falsa” en Localhost a cuentas de correo reales.

Aspectos significativos cubiertos

Desgranando punto por punto, yo diría que son las siguientes:

1. **Geoposicionamiento de Google.** Me gusta mucho como quedó la parte de mostrar resultados dinámicamente extraídos de la base de datos en Google Maps, pero sobre todo, aunque no es una parte de código muy grande, creo que es bastante significativo conseguir la geoposición (latitud / longitud) de una dirección introducida por el usuario a mano. Era una parte indispensable sobre la que se basa el proyecto y no podía renunciar a ella.
2. **Integración de fotografías en la web**. Otra parte que es bastante significativa es como trata el servidor las imágenes subidas por el usuario. Están almacenadas en el mismo servidor, con una estructura que genera una carpeta por cada animal ingresado. Se identifica con un ID único, para evitar errores de que dos usuarios suban una foto llamada igual. Y aparte, se almacena una ruta relativa en la base de datos. Para visualizarla se muestran en dos formas. La primera imagen de todas las que ha subido el usuario se usa como imagen principal para anunciar al animal, y al visitar la ficha se crea un *SlideShow* para visualizarlas de forma atractiva. Y por supuesto, los usuarios solo pueden usar archivos de imagen, pero tantos como deseen.

Aspectos sin cubrir

Hay varios aspectos que no he podido llevar a cabo por falta de tiempo o fallos en su implementación.

1. **Crear una mini red social.** Es una parte que me fastidia bastante, pero por motivos de tiempo no llegaba a crear una sección que integrase la comunicación de usuarios en chats privados, compartir imágenes en una especie de muro… Lo más aproximado fue introducir una comunicación integrando Slack en la web, puesto que en las FCTs tuve que generar un bot en esa aplicación y mucho código podía reutilizarlo, pero consideraba que era complicar en exceso una plataforma que quería que funcionase de forma simple, ya que debía solicitar un login en una página externa, seleccionar un canal de comunicación, etc. Haría perder la esencia de la web, "hasta mi madre sabrá utilizarla".
2. **Localizar la geoposición en tiempo real**. Realmente este punto lo llegue a implementar, fue bastante sencillo una vez que has conseguido generar un método de geoposicionamiento, pero en las pruebas de código que introducía me daba unos márgenes de distancia brutales usando datos móviles. La gracia de hacerlo es detectar tu posición rápido y no perder tiempo escribiendo calle, localidad… Si la detecto pero lo da tan erróneo que debo modificarlo, considero que no merece la pena implementarse por el momento. Con tiempo cuando pueda afinar mejor el proceso se añadirá. De momento se queda en que el usuario, al publicar un anuncio, debe escribir esos campos y en la parte de servidor genero la latitud y longitud en base a su información.
3. **Sistema de anuncios.** Los anuncios estaban pensados para anunciar exclusivamente publicidad de tu localidad y a nivel nacional. El código se ha dejado preparado para funcionar de esa forma, pero mientras se está creciendo hay pocos anunciantes, lo que me ha llevado a usar un sistema de generación aleatoria de anuncios y mostrar en cualquier provincia todos los anuncios. No es lo ideal pero conforme crezcan los clientes se puede añadir esa modalidad sin necesidad de modificar el código, simplemente cambiando un parámetro del procedimiento que accede a la base de datos.

Análisis de lo existente

La situación actual del sistema es un aplicación web sobre PHP + MySQL, usando Mercury como servidor de correo electrónico, todo ello montado en un sistema operativo Windows 10 Proffesional.

En el sistema actual, la estructura del proyecto es /htdocs/damelapatita como directorio donde almacenar la página web y /htdocs/img.

La página web es responsive y está desarrollada para funcionar correctamente en diversos navegadores.

Como puntos negativos, puede reprocharse que no se haya montado este servicio como una aplicación Android nativa para implementar mejor ideas como el geoposicionamiento, ya que cuenta con herramientas más específicas para ello y ofrece resultados muy atractivos y tienen mucho éxito hoy en día, y haberme decantado por una solución web con el trabajo que lleva gestionar que se muestre correctamente en tantos navegadores como existen hoy en día.

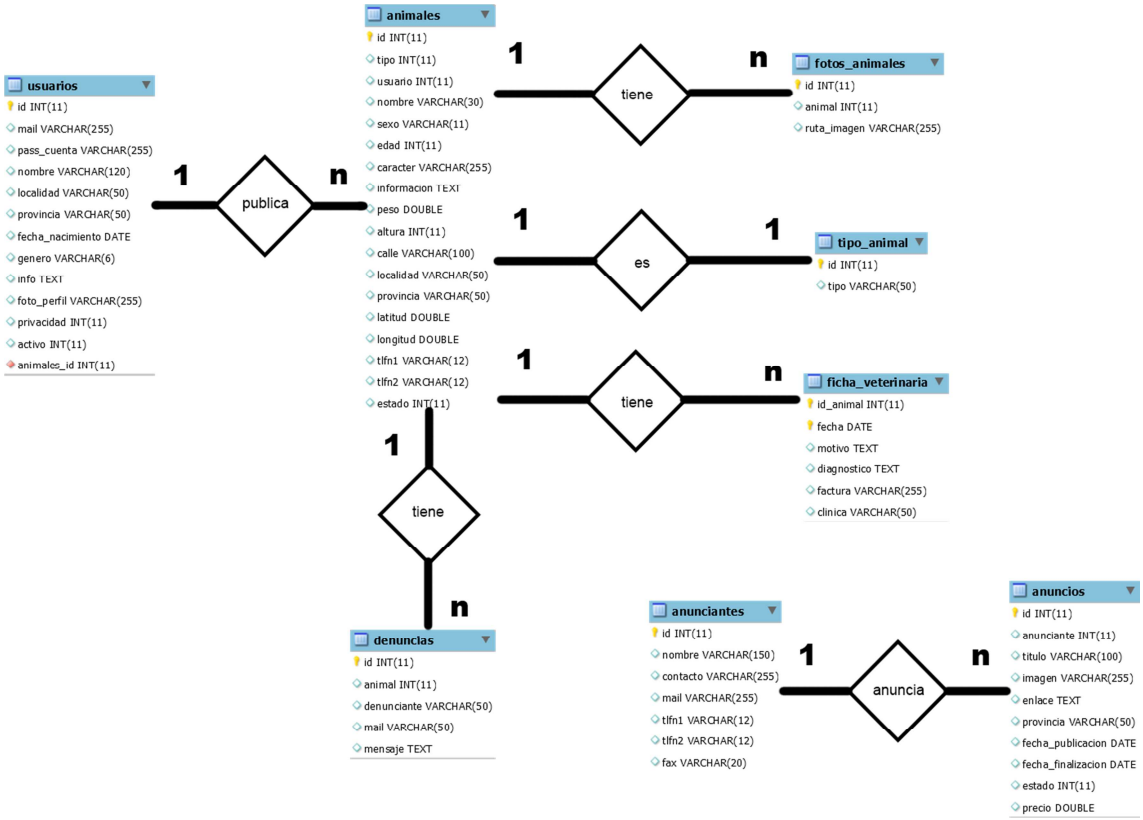
Bien, el caso es que mi solución está pensada para ser usada por el mayor número de personas. Si creo una aplicación Android estoy limitando mis usuarios a ese tipo de plataforma, obviando a los Iphone, Windows Phone, etc. Una aplicación web permite llegar a mucho más público.
Respecto al problema con los navegadores, se han usado Frameworks de desarrollo web como Bootstrap, Angular y Normalize que facilitan tremendamente la integración de la página web con los diversos navegadores web.
Evidentemente todos no se pueden cumplir, pero se ha procurado tener bien cubiertos los principales (se puede ejecutar en Chrome, Mozilla, Opera, Safari, Edge e Internet Explorer+8 sin problemas) y está habilitado para ser usado en navegadores de solo lectura como pueden ser los que incorpora Kindle y similares.

Y sí, por supuesto que hay navegadores que no están cubiertos. Por ejemplo Internet Explorer en versiones inferiores a 8 se descuadra toda la página, pero trabajando con HTML5 es de esperar, además que me parece bastante lógico centrarse en navegadores actuales y dejar de dar soporte conforme vayan quedando tan desfasados como esos.

Análisis del sistema

Diseño de la base de datos

El modelo de entidad – relación es el siguiente



Se ha elegido este modelo en concreto pensando en una futura ampliación inicial de la aplicación en la sea relativamente sencillo agregar nuevos modulos a la base de datos.

Se dividen dos situaciones claramente. Por un lado, tenemos la interacción entre los usuarios y los animales, y en otro sin conexión con ellos almacenamos los anunciantes y su publicidad.

En la sección de animales se ha decidido montar una tabla específica ANIMALES en vez de tres tablas para ADOPTADOS, PERDIDOS y ENCONTRADOS porque, pese a que se van a almacenar valores nulos en una tabla, considero que es la mejor manera de tratar los cambios de estado de un animal. Resulta más sencillo y rápido editar un campo para marcar que pertenece a una determinada categoría que no eliminar e insertar un registro en base a una simple modificación de estado.

Respecto a las tablas relacionadas con la publicidad están separadas del resto porque aunque el sistema de anuncios de la web compara la publicidad del anuncio con la del usuario, me parecía mucho más lógico hacer esa unión directamente en código que no en una base de datos. He considerado que eran caminos diferentes para adaptar la información.

Asismo la tabla ANUNCIOS tiene un evento asociado que se lanza diariamente a las 03:00 para comprobar la diferencia de fechas entre el día actual y la fecha de caducidad del mismo. Si el

conteo llega a 0 modifica el estado para que ya no sea un anuncio visible en la web, saltándoselo el sistema aleatorio de selección.

Tanto el script de creación de base de datos como el evento se adjuntan en los anexos SCRIPT1 y SCRIPT2 respectivamente.

Estructura web

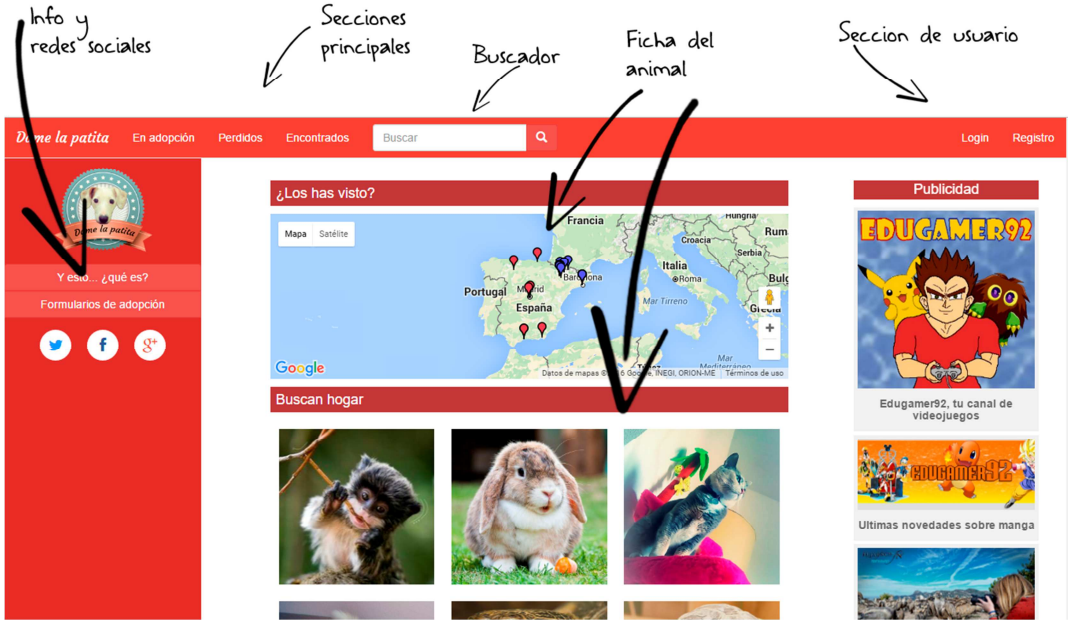
Directorio	Descripción
Damelapatita	Directorio raíz de la aplicación. Contiene index y .htaccess
Css	Contiene las hojas de estilo. Ruta de Bootstrap
misEstilos	Almacena las hojas de estilo generadas por mi
Fonts	Contiene las fuentes empleadas en la página web
Img	Imágenes usadas para el diseño de la web
Js	Ficheros Javascript. Contiene los framework de Bootstrap, jQuery, Angular y Prototype
acciones	Son los scripts de js generados por mi cuenta que interactúan directamente con el DOM
directivas	Contiene la declaración de directivas angular para generar templates
Lib	Biblioteca de funciones PHP
llamadasPHP	Ficheros PHP de interacción. El paso intermedio entre interfaz de usuario y las bibliotecas PHP y base de datos
nbproject	Carpeta del sistema para cargar en Netbeans
Paginas	Son las vistas. La interfaz gráfica que visualiza el usuario
secciones	Fragmentos de páginas reutilizables convertidos en etiquetas HTML con Angular
termsAndUses	Almaceno el fichero .txt con las condiciones de uso de la página web

Interfaces de usuario

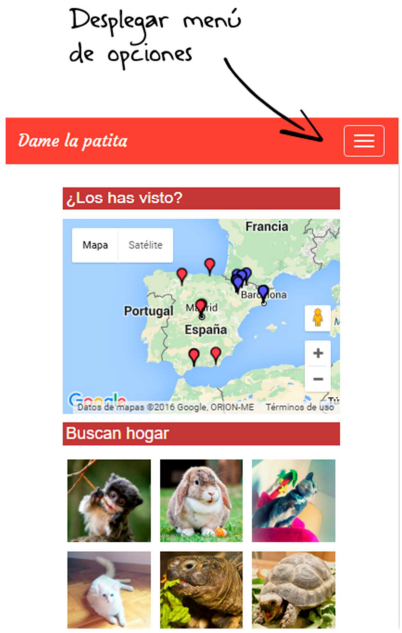
El usuario contactará con la aplicación web mediante un navegador, cargando distintas vistas dependiendo de la zona a visitar.

Si visualizamos la pantalla principal, podemos ver la interfaz con la navegación:

(ver siguiente página)



Basicamente estas son las interacciones principales que puede hacer un usuario desde la página principal. Por supuesto, la versión responsive del móvil es ligeramente distinta



Y la cabecera, una vez logueado, también redirige a nuevas vistas para el usuario

Unidades de programación utilizadas

Debido a que en el siguiente tema voy a tratar en mayor profundidad los métodos de las unidades de programación, aquí me limitaré a indicar los lenguajes y frameworks utilizados.

Front-End

- HTML5
- CSS3
- Bootstrap
- JavaScript
- jQuery
- AngularJS
- Prototype.js

Back-End

- PHP 7
- MySQL

Requisitos SW / HW

Para este proyecto no necesitamos unas herramientas de gran potencia, y salvo la licencia del sistema operativo no requiere desembolso ninguno, puedo trabajar con lo que se encuentra en el departamento.

Hardware

Requiero un ordenador con un sistema operativo Windows con 2GB de memoria, ya que XAMPP puede llegar a consumir hasta 400MB de memoria RAM si está muy sobresaturado, y una conexión a Internet por cable.

Software

Sistema operativo Windows. Indiferente la versión, pero preferible 7, 8 o 10.

XAMPP instalado con los servidores Apache versión 7, MySQL versión 5.6 y Mercury Mail, éste último correctamente configurado para permitir envíos de mails fuera de localhost. Además Apache deberá estar configurado para permitir conexiones fuera de localhost.

Proyecto en la ruta C:\xampp\htdocs, y en la misma raíz la carpeta img que contiene las imágenes subidas por los usuarios.

Las herramientas de desarrollo empleadas han sido:

- NetBeans para la programación en back-end
- Brackets para el front-end
- MySQL Workbench para los scripts SQL

El centro deberá aportarme un cable de red y asegurarme una conexión a Internet.

Diseño del sistema

Unidades de programación

En esta sección se van a listar las clases de programación empleadas, los métodos y las funciones utilizadas. Las separo por lenguaje de programación.

PHP

Conexion. Librería encargada de realizar las conexiones y desconexiones a una base de datos ya prestablecida

Métodos
conectar()
Crea un conector PDO a una base de datos MySQL
@return Conector PDO a la BD indicada, o NULL si hay algún error

desconectar()
Devuelve el conector PDO cerrado
@param $dbh Conector PDO que deseamos cerrar
@return NULL

Comunicaciones. Libreria que realiza envios de correo electronico con un emisor ya prestablecido. Se requiere configurar un servidor mail

Métodos
enviarMail($destinatario, $titulo, $mensaje)
Envia un correo electronico. Si las lineas del mensaje tienen más de 70 caracteres se cortarán a partir del 70 separandose con \r\n
@param $destinatario string. Correo electrónico del destinatario.
@param $titulo string. Titulo del email a enviar
@param $mensaje string, indiferente si es plano o enriquecido. Contenido del mail.

Formatos. Librería encargada de realizar conversiones de formatos de datos listos para ser mostrados

Métodos
cadenaEdad($meses)
Devuelve un string indicando la edad del animal. Si es mayor de once meses el resultado será "X años" y en caso contrario "X meses". Depende de la función monthToYear.
@param $meses Numero entero > 0 que hace referencia a los meses a convertir
@return string que indica la edad en meses o años

cambiarCadenasNulas($original, $siNulo)
Sustituye un string por otro en caso de que el primero sea nulo.
@param $original string a sustituir
@param $siNulo string que reemplazará a $original

@return string. $original si no era NULL, $siNulo en caso contrario

crearRutaImagen($imagen)
Devuelve la URL completa para mostrar una imagen en el servidor. La direccion del servidor se encuentra prestablecida dentro de la función.
@param $imagen string. Es la ruta de la imagen que se encuentra almacenada en la BD
@return string. La URL completa para acceder a dicha imagen

getRandomCode()
Genera un string aleatoria de seis digitos, con los simbolos [0-9][A-Z][-)(.:,;]. Se usa principalmente para generar un nuevo password aleatorio en caso de que el usuario lo haya solicitado.
El codigo está extraido de la página web
https://josemmsimo.wordpress.com/2013/01/18/obteniendo-una-cadena-alfanumerica-usando-php/.
@return string

indicarAltura($altura)
Formatea la altura del animal en un string
@param $altura Int > 0 indicando la altura del animal, o NULL en caso de no saberse
@return strig. "X cm." siendo X la altura pasada, o "Altura desconocida" si NULL

indicarPeso($peso)
Formatea el peso del animal en un string
@param $peso double > 0.0 indicando el peso del animal, o NULL en caso de no saberse.
@return string. "X kg." siendo X el peso pasado, o "Peso desconocido" si NULL

monthToYear($meses)
Convierte meses a años.
@param $meses Int > 11
@return Int indicando los años de todos los meses pasados

Geoposicionamiento. Librería usada para generar geoposicionamientos a través de la API de Google

Métodos
geocode($address)
Convierte una dirección pasada en "lenguaje humano" a latitud - longitud.
Por ejemplo, "Calle False 123, Springfield, USA" nos devolvería las coordenadas de esa dirección
@param $address es la direccion que queremos convertir a latitud - longitud, siguiendo el patrón CALLE, LOCALIDAD, PROVINCIA, PAIS, pudiendo acortar por la izquierda. Es decir, PROVINCIA, PAIS es válido pero CALLE LOCALIDAD no.
@return array de tres posiciones. 1 - latitud, 2 - longitud y 3 - string con direccion, o retorna FALSE si se ha producido algún error.

GestionAnimales. Libreria pesnada para interactuar con las tablas de animales de la Base de datos. Depende de la libreria CONEXION y trabaja conjunto a la libreria GESTION DE USUARIOS.

Métodos
actualizarEstado($idAnimal, $nuevoEstado)
Actualiza el estado de un animal.
@param $idAnimal Int. Identificador del animal que queremos actualizar
@param $nuevoEstado Int. Nuevo estado del animal

animalesMaps()
Prepara la consulta para cartgar los animales perdidos y encontrados en la sección googleMaps.php
@return $stmt Statement PDO con la consulta lista para ejecutarse

animalesAdopcion($estado, $init, $limit_end, $tipoBusqueda, $provinciaBusqueda)
Devuelve el listado de animales segun su estado (adoptado, perdido, encontrado...) con límites para permitir mostrar el resultado creando una paginación y permitiendo filtrar por tipo de animal.
@param $estado Int. Estado de los animales a listar
@param $init Int. Limite minimo de resultados
@param $limit_end Limite maximo de resultados
@param $tipoBusqueda Int. Tipo de animal a buscar
@param $provinciaBusqueda String. Provincia en la que buscar
@return Statement PDO ya ejecutado para cargar los resultados

cantidadAnialesAdoptados($estado, $tipoBusqueda, $provinciaBusqueda)
Devuelve el numero de animales de cada estado, permitiendo filtrar por tipo y provincia. Ejemplo de uso para calculos en paginacion.
@param $estado Int. Estado de los animales a listar
@param $tipoBusqueda Int. Tipo de animal a buscar
@param $provinciaBusqueda String. Provincia en la que buscar
@return Statement PDO listo para ejecutarse

cantidadAnimalesTotal()
Devuelve el ultimo ID animal en uso en la base de datos.
@return Int con el ultimo ID de animal registrado en la BD

generarDenuncia($idAnimal, $denunciante, $mail, $mensaje)
Inserta un registro en la tabla denuncias para avisar de un uso inadecuado de la aplicacion en el anuncio de un animal.
@param $idAnimal Int. Id del animal del que se hace la denuncia
@param $denunciante String. Nombre del denunciante
@param $mail String. Correo electronico del denunciante
@param $mensaje String. Motivos de la denuncia

insertarNuevoAnimal($tipoAnimal, $nombre, $usuario, $sexo, $edad, $caracter, $informacion, $peso, $altura, $calle, $localidad, $provincia, $latitud, $longitud, $tlfn1, $tlfn2, $estado)
Inserta un nuevo animal en la base de datos sin establecer ningun registro para las fotografías
@param $tipoAnimal Int. tipo del animnal
@param $nombre String. Nombre del animal
@param $usuario Int. Id del usuario que postea el anuncio
@param $sexo String. Sexo del animal

@param $edad Int >= 0. Edad del animal
@param $caracter String. Caracter del animal
@param $informacion String. Informacion sobre el animal
@param $peso Double > 0.0. Peso del animal
@param $altura Int > 0. Altura del animal
@param $calle String. Calle donde se anuncia
@param $localidad String. Localidad donde se anuncia
@param $provincia String. Provincia donde se anuncia
@param $latitud Double. Latitud de la dirección
@param $longitud Double. Longitud de la dirección
@param $tlfn1 String. Primer elefono de contacto
@param $tlfn2 String. Segundo telefono de contacto
@param $estado Int. Estado del animal

insertarFoto($idAnimal, $ruta)
Inserta un registro relacionando el id del animal con el path relativo de la fotografia.
@param idAnimal Int. Id del animal
@param ruta String. Path relativo de la imagen

listarAnimalesBusquedaCampos($estado, $palabraClave)
Devuelve todos os animales que coincidan con una palabra clave
@param $estado Int. Estado del animal
@param $palabraClave String. Palabra a buscar
@return Statement PDO listo para ser ejecutado

tiposAnimales()
Lista los tipos de animales de la BD
@return Statement PDO listo para ser ejecutado

verAnimal($idAnimal)
Lista todos los campos de un animal
@param $idAnimal Int. Id del animal
@return Statement PDO listo para ser ejecutado

verAnunciosPropios($estado, $mailUsuario)
Muestra los anuncios, por estado, que ha publicado un usuario
@param $estado Int. Estado de los animales a buscar
@param $mailUsuario string. Correo electronico del usuario que publica
@return Statement listo para ejecutar

verFotosAnimales($idAnimal)
Lista todas las fotos de un animal concreto
@param $idAnimal Int. Id del animal
@return Statement PDO listo para ejecutar

verPublicador($idAnimal)
Devuelvo el mail del usuario que publicó el animal
@param $idAnimal int. Id del animal
@return String. Mail del usuario

GestionUsuarios. Librería para interactuar con las tablas de animales de la Base de datos. Depende de la libreria CONEXION.

Métodos
actualizarNombre($mail, $nombre)
Actualiza el campo nombre de un usuario en la BD
@param $mail string Mail del usuario
@param $nombre string Nuevo nombre

actualizarLocalidad($mail, $localidad)
Actualiza el campo localidad de un usuario
@param $mail string Mail del usuario
@param $localidad string Nueva localidad

actualizarPassword($mail, $passwordAntiguo)
Actualiza el campo pass_cuenta de un usuario en la BD
@param $mail string Mail del usuario
@param $passwordAntiguo string antigua contraseña
@param $passwordNuevo string nueva contraseña

actualizarProvincia($mail, $provincia)
Actualiza el campo provincia de un usuario
@param $mail string Mail del usuario
@param $provincia string Nueva provincia

comprobarUsuario($mailUsuario)
Comprueba la existencia de un mail
@param $mailUsuario string mail a comprobar
@return Int. -2 no existe // -1 = Banneado // 0 = Dado de baja // 1 = Login permitido

encriptarPassword($password)
Encripta un password usando la recomendacion PHP7 de generar $salt automaticamente.
@param $password string Password a encriptar
@return string Password encriptado

generarPassword($mail, $pass)
Establece un nuevo password machacando el anterior. PRECAUCION: SOLO IMPLEMENTAR SI REINICIO DE PASSWORD FORZOSO. El adminstrador no tiene forma de saber cuál se ha asignado.
@param $mail string Mail del usuario
@param $pass string. Password del usuario
@return TRUE si se ha realizado el cambio, FALSE en caso contrario

loguearUsuario($mailUsuario, $passwordUsuario)
Permite a un usuario loguearse o no
@param $mailUsuario string. Mail del usuario
@param $passwordUsuario string. Password del usuario
@return int. 1 login correcto, 2 error en password, -2 no existe // -1 = Banneado // 0 = Dado de baja

nuevoUsuario($mailUsuario, $pass_cuenta, $nombreUsuario, $localidadUsuario,

$provinciaUsuario, $nacimientoUsuario, $genero)
Inserta un usuario en la base de datos
@param $mailUsuario string. Mail del usuario
@param $pass_cuenta string. Password del usuario
@param $nombreUsuario string. Nombre del usuario
@param $localidadUsuario string. Localidad del usuario
@param $provinciaUsuario string. Provincia del usuario
@param $nacimientoUsuario string. Fecha de nacimiento del usuario
@param $genero string. Genero sexual del usuario

validarPassword($passwordTest, $passwordGuardado)
Comprueba si dos contraseñas, una de ellas encriptada, coinciden
@param $passwordTest string password a comprobar
@param $passwordGuardado string password encriptado con el que validar
@return Int. 1 si coincide, 0 en caso contrario

verUsuario($idUsuario)
Devuelve el usuario que publicó el anuncio
@param $idUsuario Int. Id del usuario
@return Statement PDO listo para ser ejecutado

verUsuarioDesdeMail($mailUsuario)
Devuelve el usuario completo segun su mail
@param $mailUsuario string. Id del usuario
@return Statement PDO listo para ser ejecutado

Provincias. Librería que almacena las provincas de España y su manejo para mostrar en pantalla

Propiedades
$provincia
Array que almacena las provincias de España

Métodos
devolverProvincias()
Devuelve el array de provincias
@return Array[string] con las provincias de España

JavaScript

validaciones.js
Se encarga de realizar una llamada AJAX usando la libreria PROTOTYPE de JavaScript para comprobar si llamar a validarusuario.php y mostrar graficamente si el mail del usuario esta en uso o disponible.

jQuery

cargaPaginas.js
Carga la sección de registro en el div #barrica, sustituyendo su contenido actual

configuracion.js
Se encarga de la gestión de los botones y primera validación de los passwords dentro de la página de configuración del usuario.

El funcionamiento es el siguiente, sabiendo que tenemos un campo para la contraseña vieja, uno para la nueva y otro para repetir la nueva, y por otra parte un botón de submit. Ademas de otros input para el resto de la configuración.

De inicio, SUBMIT está habilitado. Si rellenamos algo en VIEJO, significa que queremos cambiar el password, así que si VIEJO tiene más de cero caracteres y NUEVO1 tiene 0, deshabilitamos el botón de entrada. Así nos cercioramos de que introducen un nuevo password.

Una vez introducido, si NUEVO1 y NUEVO2 coinciden, y VIEJO tiene contenido, volvemos a habilitar el botón. Si coinciden pero VIEJO no tiene contenido, se deshabilita, porque necesitamos los tres campos rellenos para cambiar el password.
Y por supuesto, si NUEVO1 y NUEVO2 no coinciden, SUBMIT se mantiene deshabilitado.

redesSociales.js
Aquí se generan los pop-up para compartir la página web en las redes sociales. Actualmente está habilitado para Twitter, Google Plus y Facebook. Whatsapp se implementa en la versión movil y solo para compartir anuncios (de momento).
La forma de compartir es generar una nueva ventana con jQuery que aproveche los metadatos cargados en la página web de la que pulsamos el botón.

registro.js
Este documento jQuery modifica el DOM de la página web usando el selector jQuery, en vez de $, porque entra en conflicto con prototype.js para la carga AJAX.
Tiene una función similar a configuracion.js. Sirve para realizar nuevos registros en la web. De entrada, SUBMIT esta deshabilitado, y solo se habilitara cuando las dos contraseñas coincidan y se hayan aceptado los términos y condiciones de uso de la página web (aceptando el check, y se pueden leer en una ventana modal).

Por otro lado, el botón limpiar campos se encarga de resetear todos los input para volver a comenzar, y volver a deshabilitar el botón de enviar.

La función comprobarPassword se encarga de devolver TRUE o FALSE si el contenido de los inputs es verdadero o falso respectivamente.

Angular

seccionesDirectivas.js
Genero una directiva angular para crear plantillas HTML (almacenadas en la carpeta secciones) que permitan dividir el código de la página web en segmentos y facilitar su reutilización.

Herramientas de programación

Como herramientas de trabajo he usado:

- **Netbeans**. El código en PHP decidi crearlo en NetBeans porque me encanta su autocomplete con este lenguaje. Probé antes con Eclipse, pero decidí migrar el proyecto porque me resultaba más cómo realizarlo aquí, además de que me lanza la ejecución en el navegador directamente y no en una ventana del propio IDE. Eso le hice ganar puntos
- **Brackets.** Para front-end no hay nada que haya encontrado mejor que este IDE. Tiene grandes funciones de autocompletado y guía, y tiene una herramienta que permite ver la página mientras la vas modificando.
- **MySQL Workbench**. La plataforma para desarrollar SQL. También, cuando solo debía modificar pequeñas cosas, lo hacía con Atom IO, pero en líneas generales he trabajado con él
- **Photoshop.** Con este software he hecho la maquetación y montaje de las imágenes. Por muy partidario del software libre que sea, sigo sin acostumbrarme a GIMP o Krita…

Medidas de seguridad de acceso a datos

Para las medidas de seguridad de la información realizo distintas tareas, según el nivel en el que trabajo.

- En la interfaz de usuario, uso sesiones para permitir o denegar el acceso a las zonas restringidas, como puede ser el panel de configuración del usuario, sus anuncios publicados, etc.
- Las contraseñas se almacenan cifradas en la base de datos, con una semilla aleatoria generada por PHP en su versión 7. Ni el webmaster puede leerla si no tiene la combinación que lo desbloquea.
- Y precisamente si el usuario la pierde, lo que se hace es machacar la antigua, generar una totalmente aleatoria y mandarla al correo con el que se registró.
- Por último, para garantizar que no hay accidentes en la administración de la base de datos, todas las funcione que ejecuta el administrador se realizan a través de proceidimientos y funciones almacenadas, para evitar fallos en la línea de “DELETE FROM” sin añadir “WHERE”.

Presupuesto (contando el plan de empresa)

Vayamos paso por paso. Dentro de esta sección voy a desgranar la creación de la empresa, los gastos que me acarrearía montar una, y posteriormente los gastos que me ha llevado, tanto de tiempo como económicos, la creación de la web, desde el punto de vista empresarial.

Creando la empresa

Una empresa que se dedique solo a gestionar esta aplicación web tiene más bien poco futuro, así que mi planning pasa por generar una empresa de desarrollo web autónoma sin un local fijo para la reducción de costes. Y dentro de esta empresa, hay una sección en la que se implementa *Dame la patita* como uno de nuestros servicios.

La empresa mantendría contactos vía online y telefónico, trabajando yo desde casa, y generando meetings con los clientes en oficinas que habiliten espacios por horas. He visto esa idea en QV Offices, donde he realizado las prácticas, y me ha parecido bastante buen plan para evitar gastos superfluos y centrarse en las soluciones web.

Tengo menos de treinta años, no soy administrador de una sociedad mercantil, ni he estado dado de alta como autónomo ni como autónomo colaborador, de modo que cumplo todos los requisitos para que se me conceda la tarifa plana para la cuota de autónomos, lo que supondría pagar los primeros seis meses 50€ y los seis siguientes 133 euros. A partir del año, y hasta alcanzar el año y medio obtendría una reducción del 30% de la cuota de autónomo, que serían 187€. Pero para las cuentas me voy a centrar solo en el primer año, para realizar una muestra de ingresos y gastos.

En resumen, pagar la cuota de autónomo me supondría un gasto el primer año **1098 euros**.

Como voy a realizar el trabajo desde casa y no necesito un local fijo donde establecer mi negocio, alquilaré una oficina de servicios por horas. Simulando datos en webs de alquier de oficinas, me sale una media aproximada de 30€ por hora de meeting, lo que deberemos multiplicar por cada vez que decida reunirme con un cliente en esa oficina.

Para nuestro primer cliente, haciendo una simulación de las reuniones que he podido ver en la empresa, pongamos que nos hemos reunido unas cinco veces en persona, con un total de 10 horas de reunión. El resto se hizo por teléfono y correo electrónico que no conlleva gasto. Así que el desembolso del local se ha quedado en **300 euros**.

Contando el local y la cuota de autonómo, debo centrarme ahora en los materiales que usare para trabajar. Necesito:

- ➔ Un ordenador con conexión a Internet. Eso está cubierto porque en mi vivienda ya poseo ambos, y me procuraré una oficina con conexión Wi-Fi para las simulaciones delante del cliente. Gasto: **0 euros**. Bueno, realmente la luz y la conexión a Internet, pero estando así en mi casa no considero que deba tratarlo como gastos. Si no debería incluir todos los gastos de una casa y sería como hacer cuentas por duplicado).
- ➔ Un servidor web donde alojar las páginas. Contratando el plan Empresarial de Hostinger el pago por mantener el servidor web es de 5.99€ al mes. En un año gastaré **71.88 euros**.

- ➔ Material de oficina. Cuadernos, posits, bolígrafos, CD/DVD, pendrives, sello… Se realiza una reserva de **150 euros** para cubrir material a lo largo de un año.

Ese sería un listado del gasto que conllevaría comenzar la rodadura de la empresa. Veámoslo más detalladamente. Recuerda que estos gastos están basados en el primer año contable de la empresa.

CREACIÓN DE LA EMPRESA	
MOTIVO	**GASTO / AÑO**
Cuota de autónomo	1098 €
Alquiler de local por horas	300 €
Ordenador + conexión a Internet	0 €
Servidor web	71.88 €
Material de oficina	150 €
	TOTAL: 1619.88 euros

Evidentemente con este gasto tan superfluo la empresa no llegará muy lejos. Necesito darme a concer.

Aquí es donde entra la segunda parte del presupuesto, el que he orientado para la publicidad. Mi gasto será:

- ➔ Desarrollo de una página web que anuncie el servicio de mi empresa. La realizo para mi propio uso, y el servidor web ya está pagandose, así que solo tendré en cuenta la compra del dominio. Consultando en *1&1* un dominio para mi empresa que sea algo como *inazense.com* costaría **0.99 euros** al año
- ➔ Tarjetas de visita. 50 tarjetas de visita clásica (84mm x 55mm), en papel reciclado (la huella ecológica es importante para mi) y con diseño personalizado me costarían 15.48 euros en MOO.com. Suponiendo que encargue un primer envío de 200 tarjetas, el precio total sería de **61.92 euros**.
- ➔ Trípticos. Aparte de las tarjetas, una buena forma de promocionar es con el uso de trípticos. Encargadas en imprenta online, 250 trípticos cuestan **73.55 euros**.

Si visualizamos precios en la tabla, podemos ver que el gasto queda así

A eso debo añadir la compra del dominio, que aunque se puede cargar al cliente, deberé desembolsar yo el dinero. El dominio que deseo comprar es *damelapatita.org*, que conlleva un gasto de **7.99 euros** al año.

PUBLICIDAD	
MOTIVO	**GASTO / AÑO**
Dominio página web empresarial	0.99 €
Tarjetas de visita	61.92 €
Trípticos	73.55 €
	TOTAL: 136.46 euros

Desarrollo de la página web

El coste del desarrollo de la página web lo voy a clasificar en dos secciones diferencidas, básicamente, por el dinero. Qué puedo cobrar y qué no.
Empezando por lo que NO puedo cobrar, ahí entra mi autoaprendizaje para aprender a manejar la tecnología necesaria para crear páginas web.

En términos de horas, estuve un mes completo, alrededor de unas cuatro horas diarias de media aprendiendo sobre las tecnologías mencionadas en las secciones anteriores. Esto me da un total de **120 horas** de formación que en mi opinión no entran como un gasto económico.

Una vez adquirida la base, el desarrollo del proyecto completo me costó una semana y media de planteamiento logístico, un mes de codificación y otras dos semanas para la depuración de errores y mejora de funcionalidades que se detectaron. Contando que de media hacía unas tres horas, y obviando muchos fines de semana y festivos – porque sí, hay que hacer el proyecto, pero estando en Inglaterra turismo era mi segundo apellido también – voy a dejarlo en 45 días de desarrollo. Multiplicando, nos da la cantidad de **135 horas**.

El problema que se me plantea es que desconozco cuánto cobra un programador por realizar una página web, cosa de que las FCTs sean de gratis, así que suponiendo que esté algo mejor pagado que el puesto de técnico informático, vamos a poner un valor de la hora trabajada de ocho euros, con lo que el coste para la empresa sería de **1080 euros**. Bueno, imaginemos que el programador es junior en ese caso.

Ahora bien, otra cosa distinta es a cuanto cobraría la empresa la hora tabajada al cliente. De nuevo tiro de referencia. Una hora en mi anterior trabajo costaba al cliente 38 euros + IVA. Si imaginamos que tiene precios bajos por ser nueva en el mercado, podríamos decir que la hora la cobrará a 20 euros + IVA, lo que le llevaría un gasto al cliente de **2848.50 euros** (sin embargo, se obviará para el cálculo porque en este caso soy yo mismo).

Y aparte de eso, hay que comprar el dominio para colgar la web. Si nos decantamos por *damelapatita.org*, en *1&1* el precio es de **7.99 euros** al año.

Entonces, si visualizamos la información en la tabla, nos queda lo siguiente:

PÁGINA WEB		
MOTIVO	**GASTO (TIEMP)**	**GASTO (ECONÓMICO)**
Formación en tecnologías	120 h.	0 € (el aprender no tiene precio)
Desarrollo de página web	135 h.	1080 €
Dominio web	0	7.99 €
TOTAL:	255 horas	1087.99 euros

Presupuesto total

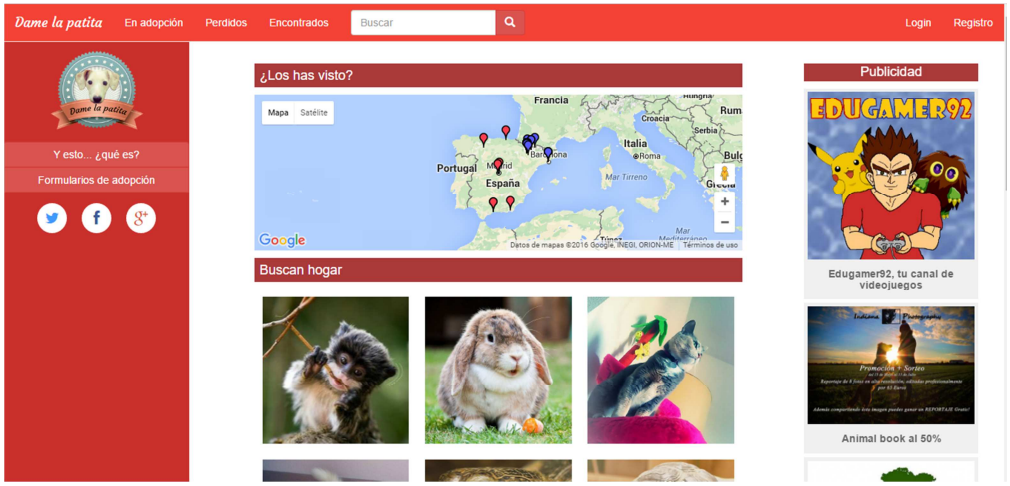
PRESUPUESTO FINAL	
MOTIVO	**GASTO / AÑO**
Creación de la empresa	1619.88 €
Publicidad	136.46 €
Página web	1087.99 €
	TOTAL: 2844.33 euros

Manual de usuario

Descripción del sistema

Pantallas superiores a 871px

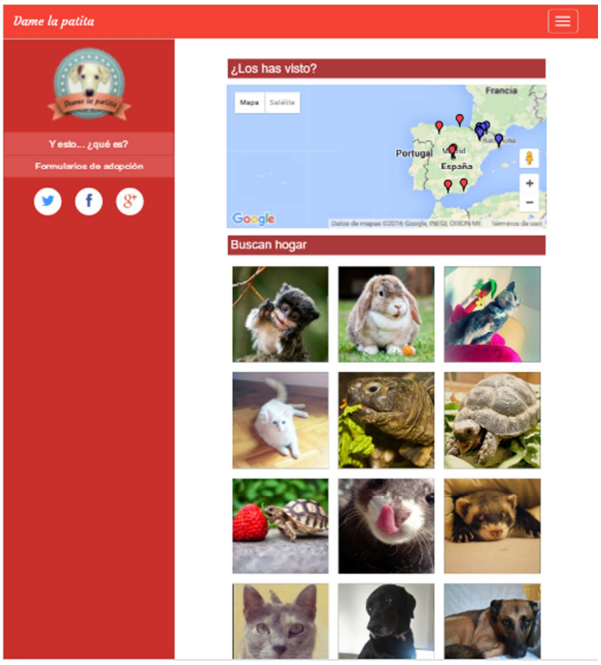
Nada más entrar en nuestra aplicación nos podemos encontrar la siguiente pantalla



Tenemos cuatro secciones claramente diferenciadas.

- **Cabecera.** Desde aquí accederemos al resto de secciones de la web, así como loguearnos y registrarnos
- **Sidebar (panel) izquierdo.** Una pequeña introducción a la página web, habilitar un formulario de descarga y compartir la página en redes sociales.
- **Sidebar (panel) derecho.** Anuncios publicitarios. Pulsando sobre ellos podemos ir a la página web que promocionan.
- **Sección central.** Esta es la zona principal de la página inicial. En la parte superior tenemos los animes perdidos publicados en un mapa de Google Maps. Hay dos colores para distinguirlos. Azul para los que se han perdido, y rojo para aquellos que la gente se ha encontrado por la calle. Y luego la zona inferior, donde visualizo los doce últimos animales publicados. Cuando posicionamos sobre una imagen, esta se oscurece y muestra datos principales del animal. Además, al pulsar abre su ficha completa

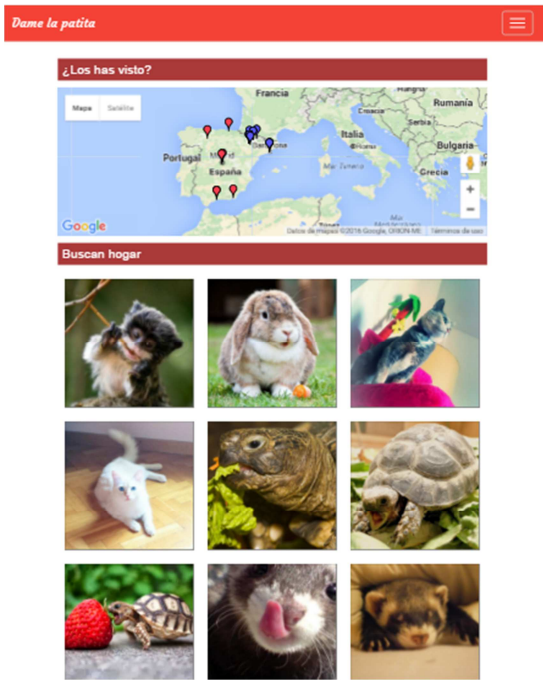
Pantallas inferiores a 872px



Se mantienen las mismas funcionalidades que en la versión de escritorio pero el aspecto es ligeramente diferente. La sección publicitaria se ha trasladado al final de la sección central y la cabecera se ha ocultado haciéndola desplegable.

Aparte de eso, la página se mantiene igual.

Pantallas inferiores a 768 px

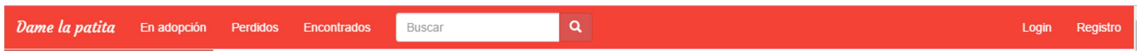


En esta vista se siguen ocultando elementos. En este caso seguimos mostrando la publicidad en la sección inferior a la central, pero ocultamos del todo el menú izquierdo con la información, ya que su principal atractivo, facilitar el formulario de adopción, no tiene mucho sentido usarlo desde el móvil cuando su lectura y modificación se empobrece considerablemente.

Aparte, el menú pasa a ser completamente vertical al ser desplegado:

Manejo de los menús

El funcionamiento, pese a presentar un estilo distinto, es el mismo dando igual el dispositivos donde lo visualicemos, así que para esta explicación usaremos la primera vista, para pantallas grandes.



En esta imagen podemos diferenciar claramente varias opciones del menú principal.

- **Dame la patita.** Siempre que pulsemos sobre ella, nos redireccionará a la pantalla principal
- **En adopción.** Muestra, al igual que en la sección central, todos los animales que están en adopción
- **Perdidos.** Lo mismo que *En adopción* pero con los animales que se han reportado como perdidos
- **Encontrados.** Lo mismo, pero con aquellos que los usuarios se han encontrado por la calle.
- **Buscar.** Nos permite buscar cualquier término sobre los animales. Busca coincidencias de palabras en la base de datos. NO BUSCA INFORMACIÓN SOBRE LOS USUARIOS.
- **Login.** Genera una ventana modal en la que identificarnos.
- **Registro.** Abre una sección central donde poder registrarnos

Una vez que nos hemos identificado, el menú cambia para permitir también la gestión de tus datos y la publicación de datos.

La principal diferencia es haber eliminado *Login* y *Registro* y haber agregado

- **Publicar**. Publica un anuncio en la web
- **Mi zona**. Al pulsarla, despliega varias opciones. A saber

- **Mis animales**. Nos abre una zona para mostrarnos todos nuestros anuncios, y permitir modificarlos, borrarlos o darlos como solucionados.
- **Configuración**. Abre una sección para cambiar nuestros datos de usuario
- **Cerrar sesión**. Nos desloguea de la aplicación y volvemos al punto de partida.

Gestión periódica del sistema

Un evento recorre la base de datos automáticamente a las 03:00 am para actualizar los anuncios, y otro script genera los backups diariamente también automáticamente.

Manualmente nos deberemos preocupar por gestionar los avisos de denuncias, baneos y limpiar la base de datos una vez al mes con todos aquellos registros que hayan solicitado el borrado de información, además de publicar los anuncios.
Para ello se realizará una conexión a la base de datos mysql y modificarla a través de consola de comandos.

Gestión de copias de seguridad

Se realiza una copia de seguridad diariamente a las 03:00 am.
Se ha generado este script:

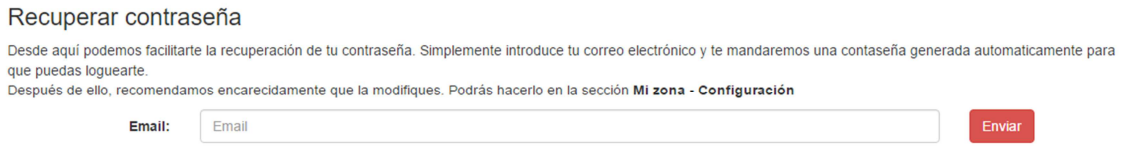
```
mysqldump.exe --user=root --password=root --host=localhost --port=3306 --result-file="F:\backup.%date:~10,4%%date:~7,2%%date:~4,2%.sql" --default-character-set=utf8 --single-transaction=TRUE --databases "proyecto_fin_dam"
```

Y se ha programado una tarea automática en el Windows Schedule Task. De esta forma nos desentendemos de las copias de seguridad manuales.

Mensajes de error

En la aplicación se deben controlar varios errores. El primero, que la página no exista. Error 404. Cada vez que se lance ese caso, nos redireccionará automáticamente aquí

Otro error común es haber olvidado el usuario a la hora de loguearnos. De ser el caso, se ha habilitado un enlace que nos llevará a la siguiente dirección:



Con indicaciones posteriores para modificar tu contraseña una vez logueado.

Y también deberemos controlar que no se quieran registrar dos veces con el mismo mail, cosa que hacemos desde la sección de registro comprobando el correo con Ajax.

Ya existe:

Sin registrar:

Bibliografía

Consultas y documentación técnicas

Autor	Titulo	Web
FalconMasters	Curso de Bootstrap 3	https://www.youtube.com/playlist?list=PLhSj3UTs2_yWTKvu1Aq3xUhzIJNBZ3MFW
---	Manual de PHP	http://php.net/manual/es/index.php
W3Schools	Google Maps API Tutorial	http://www.w3schools.com/googleapi
CodeSchool	Learn to build an application using AngularJS	http://campus.codeschool.com/courses/shaping-up-with-angular-js
Microsoft	Programar una tarea	http://windows.microsoft.com/es-419/windows/schedule-task#1TC=windows-7

Autor(es)	Título	Editorial	Año publicación
Eric T. Freeman y Elisabeth Robson	Head First: Javascript	O'Really	2014
Dan Pilone y Russ Miles	Head First: Software Development	O'Really	2007

Consultas y documentación no técnicas

Autor	Titulo	Web
1&1	Comprobar dominios	https://www.1and1.es
Hostinger	Hosting web	http://www.hostinger.es/hosting-web
InfoAutonomos	Tarifa plana para autónomos, jóvenes y mayores de 30	http://infoautonomos.eleconomista.es/seguridad-social/tarifa-plana-autonomos-50-euros-mayores-30-jovenes/

ANEXO I. Script de creación de la base de datos

```
/**
 * Author:  Inazio
 * Created: 29-abr-2016
 */

-- Creo la base de datos
drop database if exists proyecto_fin_dam;
create database proyecto_fin_dam;

-- Uso la base de datos
use proyecto_fin_dam;

-- Borro las tablas previas
-- (por si no eliminase la BD)

drop table if exists usuarios;
drop table if exists animales;
drop table if exists tipo_animal;
drop table if exists animales_foto;
drop table if exists ficha_veterinaria;
drop table if exists anunciantes;
drop table if exists anuncios;
drop table if exists denuncias;
-- Creo las tablas

create table tipo_animal(
    id int auto_increment primary key,
    tipo varchar(50)
);

create table usuarios(
    id int auto_increment primary key,
    mail varchar(255) unique,
    pass_cuenta varchar(255),
    nombre varchar(120),
    localidad varchar(50),
    provincia varchar(50),
    fecha_nacimiento date,
    genero varchar(6),
    info text,
    foto_perfil varchar(255),
    privacidad int, -- 0 = Publico // 1 = Ocultar mail // 2 =
mail + fechaNacimiento // 3 = mail + fecha + genero // 4 = mail
+ fecha + genero + localidad + provincia
    activo int -- -1 = Banneado // 0 = Dado de baja // 1 = Login
permitido
);

create table animales(
    id int auto_increment primary key,
    tipo int references tipo_animal(id),
    usuario int references usuarios(id),
```

```
    nombre varchar(30),
    sexo varchar(11),
    edad int,
    caracter varchar(255),
    informacion text,
    peso double,
    altura int,
    calle varchar(100),
    localidad varchar(50),
    provincia varchar(50),
    latitud double, -- Para google maps
    longitud double, -- Para google maps
    tlfn1 varchar(12),
    tlfn2 varchar(12),
    estado int -- 0 = solicitud de borrado // 1 = familia //2 =
en adopcion // 3 = perdido // 4 = encontrado
);

create table fotos_animales(
    id int auto_increment primary key,
    animal int references animales(id),
    ruta_imagen varchar(255)
);

create table ficha_veterinaria(
    id_animal int references animales(id),
    fecha date,
    motivo text,
    diagnostico text,
    factura varchar(255),
    clinica varchar(50),
    primary key (id_animal, fecha)
);

create table denuncias(
     id int auto_increment primary key,
    animal int references animales(id),
    denunciante varchar(50),
    mail varchar(50),
    mensaje text
);

create table anunciantes(
    id int auto_increment primary key,
    nombre varchar(150),
    contacto varchar(255),
    mail varchar(255),
    tlfn1 varchar(12),
    tlfn2 varchar(12),
    fax varchar(20)
);

create table anuncios(
    id int auto_increment primary key,
    anunciante int references anunciantes(id),
```

```
    titulo varchar(100),
    imagen varchar(255),
     enlace text,
    provincia varchar(50),
    fecha_publicacion date,
    fecha_finalizacion date,
    estado int, -- 0 = Caducado // 1 = <= 10 dias para caducar
// 2 = Anunciado
    precio double
);

-- USUARIOS
create user 'usuario_basico'@'%' identified by 'm0nster!!';

-- Permisos de seleccción
grant select on proyecto_fin_dam.* to 'usuario_basico'@'%';

-- Permisos de inserción
grant insert on proyecto_fin_dam.usuarios to
'usuario_basico'@'%';
grant insert on proyecto_fin_dam.animales to
'usuario_basico'@'%';
grant insert on proyecto_fin_dam.fotos_animales to
'usuario_basico'@'%';
grant insert on proyecto_fin_dam.ficha_veterinaria to
'usuario_basico'@'%';

-- Permisos de actualización
grant update on proyecto_fin_dam.usuarios to
'usuario_basico'@'%';
grant update on proyecto_fin_dam.animales to
'usuario_basico'@'%';
grant update on proyecto_fin_dam.fotos_animales to
'usuario_basico'@'%';
grant update on proyecto_fin_dam.ficha_veterinaria to
'usuario_basico'@'%';

-- Refresco los privilegios
flush privileges;
```

ANEXO II. Evento SQL ActualizarEstado

Este evento actualiza el estado de los anuncios diariamente a las 03:00

```
set global event_scheduler = on;

create event e_ActualizarEstado
on schedule every 1 day starts '2016-05-22 03:00:00'
do call actualizar_estado()
```