ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación :

INGENIERO TÉCNICO DE TELECOMUNICACIÓN, ESPECIALIDAD EN SONIDO E IMAGEN

Título del proyecto:

DESARROLLO DE UNA APLICACIÓN WEB PARA SMARTPHONES SOBRE LOS CENTROS DE SALUD PROPORCIONADOS POR OPEN DATA NAVARRA

Pablo Eugui Arrizabalaga

Marko Galarza Galarza

Pamplona, 22 de Febrero 2013

INDICE

1. OBJETIVOS

El objetivo principal de nuestro proyecto es combinar la nueva iniciativa del Gobierno de Navarra llamada Open Data, que permite a los ciudadanos tener acceso a bases de datos de la Comunidad Foral de todo tipo, con el estudio de un marco de trabajo para crear aplicaciones web para dispositivos móviles.

Nos hemos fijado que Open Data Navarra ofrece muchas posibilidades para las personas emprendedoras aunque todavía no conocemos muy bien su funcionamiento ni sus ventajas y limitaciones pero es una herramienta que ha puesto la Administración Pública a disposición de todos los ciudadanos y empresas y es por esto que vamos a intentar sacarle partido y de paso profundizar en ella para conocer todo lo que ofrece.

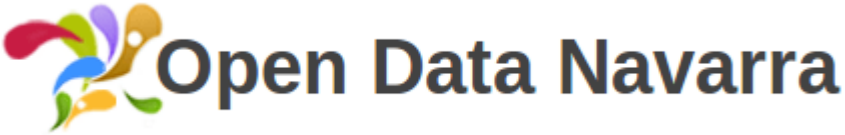
Para esto hemos decidido realizar una aplicación web para dispositivos móviles ya que es uno de los sectores tecnológicos que están en auge estos últimos años con la llegada de los smartphones. El sector de las telecomunicaciones es el segundo que más dinero mueve del mundo tras el petróleo y dentro del sector de las telecomunicaciones todo lo relacionado con los dispositivos móviles como aplicaciones es algo que está revolucionando el mercado ya que existe una gran demanda de estas aplicaciones al haberse impuesto los smartphones en la vida de las personas, siendo incluso indispensables para algunas.

Existen numerosos proyectos de fin de carrera basados en la creación de una aplicación web, el problema es que todos usaban un framework llamado JQuery Mobile, que es el más desarrollado ya que fue el primero en permitir desarrollar este tipo de aplicaciones. El impedimento que nos encontramos aparte de que era algo que ya estaba hecho es que dicho framework a pesar de ofrecer muchas posibilidades trabajaba de forma muy lenta, contiene ficheros muy pesados lo que hace más lenta la carga de las páginas web y a su vez tiene algunas limitaciones en cuanto a funcionalidad. Por esto no es convincente hacerlo con este framework ya que hace unos años era lo más conveniente ya que era el primero y el más desarrollado. Actualmente con el paso del tiempo han ido apareciendo nuevos frameworks similares que todavía no son tan famosos ya que JQuery Mobile se ha hecho mucho renombre, pero esto no indica que sea el mejor.

Por lo tanto vamos a proceder a investigar 3 nuevos frameworks disponibles en el mercado para realizar aplicaciones web y que ofrezcan una alternativa a utilizar JQuery, finalmente elegiremos uno de estos 3 que sea el que más nos convenza y procederemos a realizar una aplicación web con él combinándola con OpenData Navarra.

A continuación introduciremos y profundizaremos en todos estos aspectos y fundamentos para poder entender mejor como se ha realizado el presente proyecto.

2. OPEN DATA NAVARRA



Es una iniciativa mundial que pretende que los datos e información de las Administraciones Públicas se expongan y sean accesibles de forma que estén disponibles para su redistribución, reutilización y aprovechamiento por parte de los ciudadanos y las empresas para conseguir un beneficio para todas las partes.

Tener acceso a los datos de la Administración garantiza la transparencia porque se tiene acceso a datos que proceden directamente de fuentes oficiales. También se fomenta la eficiencia y la igualdad de oportunidades, ya que los ciudadanos y las empresas pueden crear servicios que resuelvan sus necesidades en colaboración con la Administración y todo el mundo puede acceder a los datos en igualdad de condiciones.

La apertura de datos revierte directamente en la sociedad navarra beneficios tanto en términos económicos como de conocimiento. Para las empresas y el sistema económico en general presenta un potencial económico y una oportunidad de generación de riqueza, ya que pueden utilizar la información como base para crear nuevas herramientas y servicios digitales innovadores que ofrecer a los ciudadanos y a otras empresas. Por lo tanto, también puede ser un impulso para el crecimiento y la creación de empleo. Por otra parte, la colaboración entre empresas puede acarrear una reducción de costes.

En noviembre 2011 se publica una primera versión del Catalogo de Productos. Es un sistema de información que permite catalogar los productos software de la Dirección General de Gobierno Abierto y Nuevas Tecnologías y que dan servicio a distintas unidades de Gobierno de Navarra. El proceso de catalogación no está terminado, por lo que se actualizará; desde la propia página web se puede descargar el documento en pdf, y en la ficha de datos el dataset en diversos formatos como xml, csv o json.

Open Data ofrece una gran oportunidad de negocio a empresas y personas emprendedoras. Un reciente estudio realizado por el ONTSI (Observatorio Nacional de las Telecomunicaciones y la Sociedad de la Información) acerca del sector infomediario sobre esta nueva herramienta indica que de hecho, la reutilización de la información pública en España ya genera un volumen de negocio anual de entre 550 y 650 millones de euros y el sector emplea directamente a cerca de 5.500 trabajadores.

El informe define como empresas infomediarias al conjunto de compañías que generan aplicaciones, productos o servicios de valor añadido destinados a terceros, a partir de la información del sector público; y se han identificado a 230 empresas, catalogadas en subsectores, en función del ámbito de información que reutilizan: Negocio/ Económico, Jurídico/ Legal, Geográfico/ Cartográfico, Meteorológico, Socio demográfico/ Estadístico y de Transportes.

Casos de éxito

Entre las empresas españolas destacan algunos casos como el de Euroalert.net, que ofrece acceso diario a información, contenidos y servicios relevantes relacionados con la Unión Europea, en español y en inglés. Euroalert está trabajando en un proyecto que pretende construir un prototipo de plataforma europea que agregue todos los concursos públicos de los 27 estados miembros de la UE, con el fin de diseñar productos y servicios de información baratos y accesibles que ayudarán a las pymes a ser más competitivas en este mercado, lo que pondrá a disposición de pequeñas y medianas empresas de toda Europa oportunidades por valor del 17% del PIB de la UE.

Otro ejemplo de reutilización de datos es Licitaciones.es. Su sistema les permite introducir todos los concursos públicos que puedan recoger y hacer búsquedas filtradas mediante palabras clave, que se adaptan a cada cliente, de manera que las pymes no tengan que invertir horas en la lectura de decenas de boletines y perfiles de contratante para encontrar las licitaciones que encajan con sus líneas de negocio. Licitaciones.es está triplicando en 2011 la facturación media del año anterior.

En el caso de ITEISA, recupera, procesa, organiza y relaciona desde 2007 los precios públicos de los carburantes en más de 8.000 estaciones de servicio españolas. ITEISA provee esta información a dos multinacionales extranjeras para la integración en sus sistemas GPS y pone a disposición del público un buscador online sobre el precio de la gasolina. La empresa también gestiona un servicio de vigilancia sobre boletines oficiales que reciben diariamente cerca de 600 destinatarios.

El Gobierno de Navarra está fomentando el uso de este nuevo proyecto abierto al público mediante concursos y premios. En 2011 OpenData realizó un concurso sobre aplicaciones móviles utilizando estos datos de la Comunidad Foral con el fin de promover y promocionar su uso. Se pueden ver los ganadores del concurso en la propia página de OpenData:

http://www.navarra.es/home_es/Open-Data/

Por estos motivos decidimos coger los datos de este nuevo proyecto impulsado por el Gobierno de Navarra, ya que de esta manera los datos son oficiales y a su vez apoyamos una iniciativa que favorece a todo ciudadano de la Comunidad Foral.

Escogimos una base de datos en la que se recogían todos los centros de salud y hospitales de Navarra, así como su dirección, teléfono de contacto, localidad, tipo de centro y más datos. Se pretende realizar una aplicación web que permita al usuario saber donde están los centros de salud, su teléfono para pedir cita o incluso encontrar el más cercano a la posición donde esté el usuario.

Una vez explicada y argumentada la elección sobre el tema que tratará nuestra webapp podemos pasar a explicar como la hemos desarrollado y creado.

3. CONCEPTOS BÁSICOS

3.1 Qué es una webapp

Una aplicación web, también llamada 'web app', es una aplicación que es accesible por los usuarios a través de un servidor web, ya sea a través de Internet o mediante un navegador. Es decir, es una aplicación de software codificada para que pueda ser leída mediante un navegador web, de manera que sea este el que realice la ejecución de la propia aplicación.

Lo que hace populares las aplicaciones web es lo práctico que es utilizar el servidor web como cliente así como la facilidad para actualizar o modificar la aplicación sin necesidad de distribuir a cada usuario el nuevo software modificado, ya que no es el usuario el que posee la propia aplicación, sino que accede a ella mediante un navegador web.

De esta manera, la principal ventaja que ofrecen las aplicaciones web es que tanto el cliente como el servidor o el protocolo utilizado no necesitan ser creados y diseñador por el realizador de la aplicación, ya que se utilizan los que están ya estandarizados. Para comprender mejor estos conceptos es necesario definir cliente y servidor:

-El *cliente* es una aplicación informática o dispositivo que consume un servicio de un computador remoto al cual se llama servidor. El objetivo que tiene el cliente web es interpretar las páginas HTML y los diferentes recursos que las componen así como los códigos y recursos que las componen (HTML, CSS, Javascript…).

-El *servidor web* es un programa informático que contiene y procesa una aplicación y que está permanentemente esperando solicitudes de conexiones con el cliente generando una respuesta a la petición de este. Está formado por diferentes elementos como los documentos HTML, scripts, recursos y documentos adicionales que son empleados dentro de las páginas que se ejecutan en el servidor cuando el cliente los solicita.

3.2 Aplicaciones web frente a aplicaciones nativas

Las aplicaciones para dispositivos móviles se pueden dividir en dos tipos diferentes. Las aplicaciones web explicadas anteriormente y las aplicaciones nativas o native apps, es decir, las que se quedan instaladas de manera permanente en el dispositivo como un software más. Existen diversas opiniones acerca de cuáles son mejores ya que depende de la función que busquemos en cada una de ellas pero todo queda reducido a la elección del cliente y a lo que mejor le conviene así como a observar cómo se adapta a sus necesidades y comodidades.

Antes de elegir uno de estos dos tipos debemos observar tres aspectos importantes:

-El coste: El objetivo al realizar una aplicación web (por no decir el objetivo principal de todo el mundo) es minimizar los costes y buscar la alternativa más económica siempre y cuando se adapte a nuestras necesidades. Es una elección difícil ya que una aplicación nativa sería mucho más cara pero depende de los recursos que se necesiten puede ser lo más rentable. Eso sin tener en cuenta la complejidad de una aplicación nativa frente a una webapp.

-La conectividad: Hay que tener en cuenta que muchas veces no se puede tener conexión a internet, por lo que si necesitamos utilizar la aplicación en uno de estos lugares sería indispensable decantarnos por una aplicación nativa ya que una vez instalada no sería necesaria dicha conexión.

-El conocimiento del cliente: Esto es imprescindible para tener en cuenta ya que si el cliente puede tener diferentes tipos de dispositivos móviles sería más conveniente realizar una aplicación web ya que no es necesario programarla para los distintos sistemas operativos ya que se ejecuta en el navegador.

Teniendo estas ideas en mente vamos a explicar brevemente las ventajas e inconvenientes de las webapps, ya que finalmente nos hemos decantado por utilizar este tipo de aplicaciones.

VENTAJAS

- Compatible con todos los sistemas operativos.

- Una vez cargada la aplicación, todos los datos están a disposición del cliente sin necesidad de cargar más apartados o complementos

- No es necesario para el cliente actualizar la aplicación ya que se actualizará automáticamente al acceder a ella en el navegador.

- No ocupa memoria en el disco duro del usuario.

- Mayor funcionalidad con la continua actualización de los diferentes navegadores.

- Movilidad ya que podemos tener acceso a ellas desde cualquier dispositivo conectado a la red.

INCONVENIENTES

- El cliente sólo dispone de la última versión, esto puede ser bueno o malo ya que si no prefiere una anterior no puede elegir.

- Necesitamos disponer de conexión a la red en todo momento, para acceder a ellas y ejecutarlas.

- Si no estamos conectados o si se interrumpe la conexión al ejecutarlas, no podremos utilizar la Web App.

- En el caso de las nativas, el cliente siempre podrá usarla mientras la tenga instalada en su dispositivo.

- Las aplicaciones nativas todavía están un paso por delante de las aplicaciones web pese al avance de HTML5 ya que no están limitadas por un navegador web.

	Ventajas	Inconvenientes
Apps Nativas	✓ Mejor rendimiento ✓ Interfaces más intuitivas ✓ Gestos multitouch ✓ Acceso a todas las funcionalidades ✓ Mayor ciclo de vida de la aplicación ✓ Acceso a iAd (y otros proveedores de Ads) ✓ Presencia en la AppStore	✓ Una implementación por cada plataforma ✓ Implementación más costosa ✓ Necesidad de personal más calificado
WebApps	✓ Implementación multiplataforma ✓ Un programador web puede desarrollarla ✓ Gracias a HTML5 y CSS3 pueden emularse la mayor parte de las animaciones	✓ Interfaces más pobres ✓ Baja reutilización de código ✓ Persistencia muy limitada ✓ Ciclo de vida de la aplicación más corto

Figura 1 - Cuadro extraído de una comparativa entre ambas http://aplicaciones-moviles.blogspot.com.es/2012/06/apps-nativas-vs-webapps.html

En nuestro caso se ha elegido crear una aplicación web ya que para lo que se propone es lo más conveniente, aparte del hecho de que realizar una aplicación nativa conlleva tener amplios conocimientos de programación e informática, por lo que para lo que queremos hacer utilizaremos un framework para hacer una WebApp.

3.3 HTML5

HTML 5 (*HyperText Markup Language*, versión 5) es la quinta revisión importante del lenguaje básico de World Wide Web. Con esta nueva versión se pretende eliminar los problemas con que se encuentran los desarrolladores web y añadir y rediseñar código adaptándolo a las nuevas necesidades de los desarrolladores.

Las nuevas mejoras que se han introducido en esta versión son:

-Mejoras en el elemento <canvas>, capaz de dibujar en los navegadores más importantes elementos 3D mediante las funciones de una API (Interfaz de Programación de Aplicaciones, es un conjunto de funciones y aplicaciones que ofrece una biblioteca para ser usados en otro software) sin necesidad de tener instalado ningún plugin o complemento.

-Cambio en la estructura del cuerpo de la web, ya que la mayoría tienen un formato común, es decir, una cabecera, un menú de navegación, pie, etc. Con HTML 5 se han añadido unas etiquetas específicas para cada una de estas partes, facilitando la programación del diseño.

-Se han eliminado etiquetas de la anterior versión de HTML que han quedado obsoletas como por ejemplo <Font> o que servían únicamente a la presentación del documento, los efectos de estas etiquetas serán manejados ahora por las hojas de estilo CSS.

-Se añaden mejoras en los formularios. Nuevos tipos de datos (eMail, number, url, datetime) y facilidades para validar el contenido sin necesidad de utilizar Javascript.

-Drag & Drop. Nueva funcionalidad para arrastrar objetos como imágenes.

-Geolocalización, ahora mediante el uso de una API se puede localizar geográficamente a los usuarios de las páginas web.

-Nuevas plantillas e interfaces de usuario, las cuales están siendo muy demandadas y se incorporan a HTML 5 para su utilización.

Como se observa, HTML 5 posee una mejor estructura que sus anteriores versiones y nos da más facilidad para el desarrollo de nuestra webapp. Además esto facilita la observación del código fuente, ya que es mucho más fácil de entender y más limpio que con otras versiones de HTML ya que se eliminan muchas etiquetas.

Veamos cuales son las nuevas etiquetas más importantes que necesitaremos usar en nuestro proyecto de webapp:

<section></section>: Como bien indica el nombre representa una sección dentro del documento o aplicación. Puede contener otras secciones dentro y se puede estructurar mejor toda la página creando jerarquías de contenido dentro de cada sección.

<header></header>: Este elemento sirve para crear una cabecera en la página. Todos los elementos que estén dentro de esta etiqueta se situarán en dicha cabecera.

<footer></footer>: Con este otro recurso se representara el pie de una sección de manera que todo lo que se introduzca en esta etiqueta quede en la parte baja de la sección.

<article></article>: Esta etiqueta podemos separar cada uno de los elementos independientes que pudieran componer un <section> concreto. Un <article> podría considerarse por tanto como cada uno de los elementos en que podemos dividir un section, o como bien dice su nombre, como cada artículo de una sección.

<aside></aside>: Representa una sección de la página que abarca un contenido poco relacionado con el que lo rodea, por lo que se le puede considerar un contenido independiente. Este elemento puede utilizarse para barras laterales, elementos publicitarios, es decir, contenidos que se consideren separados del contenido principal de la página.

<nav></nav>: Este elemento sirver para representar una sección de una página que es un enlace a páginas externas o a otras partes dentro de la página. Es algo así como un menú de navegación.

<embed>: Se emplea para el contenido insertado que necesita plugins como el Flash. Es un elemento que ya reconocen todos los navegadores.

<canvas>: Como se ha comentado antes, es un elemento complejo que permite que generar gráficos al hacer dibujos en su interior. Es un elemento muy utilizado en Google Maps, por lo que para nuestro proyecto será importante.

En esta figura se puede observar un esquema del diseño de una plantilla con estas etiquetas, para que se aprecie mejor como se sitúan en la web:

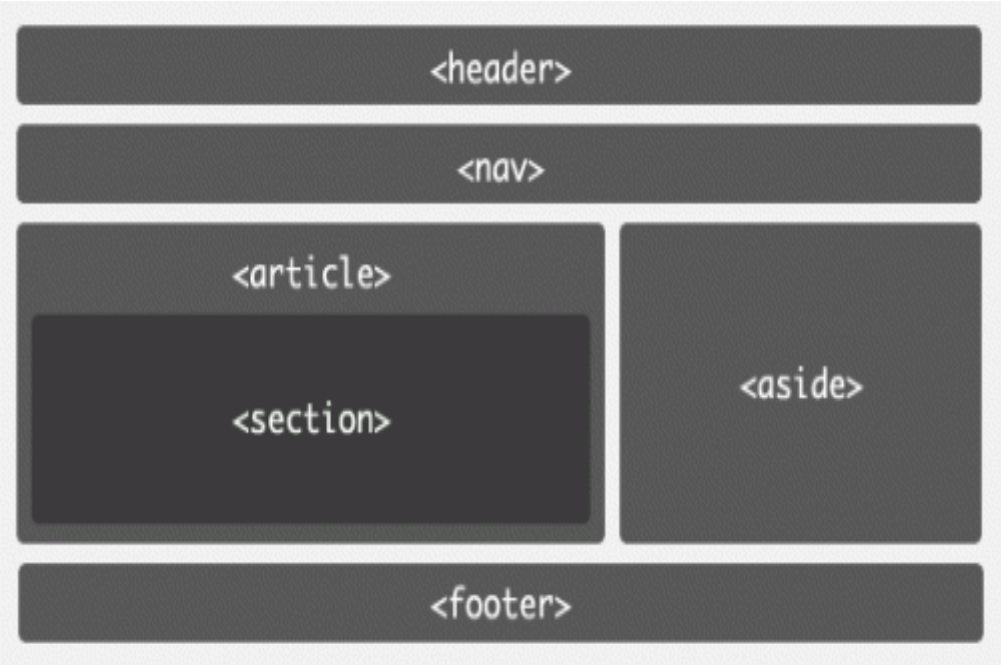


Figura 2 – Diseño básico de HTML5

Y en la figura 3 podemos ver cómo es un código en HTML5:

```
<!DOCTYPE html>

<html lang="es">

<head>
<title>Titulo de la web</title>
<meta charset="utf-8" />
<link rel="stylesheet" href="estilos.css" />
<link rel="shortcut icon" href="/favicon.ico" />
<link rel="alternate" type="application/rss+xml"/>
</head>

<body>
    <header>
       <h1>Mi sitio web</h1>
       <p>Mi sitio web creado en html5</p>
    </header>
    <section>
       <article>
           <h2>Titilo de contenido<h2>
           <p> Contenido (ademas de imagenes, citas, videos etc.) </p>
       </article>
    </section>
    <aside>
       <h3>Titulo de contenido</h3>
           <p>contenido</p>
    </aside>
    <footer>
        Creado por mi
    </footer>
</body>
</html>
```

Figura 3 – Ejemplo de código HTML5

3.4 Javascript

Este es prácticamente el lenguaje más utilizado es este proyecto. es un lenguaje de programación interpretado definido con una orientación a objetos y dinámico.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM) el cual explicamos en el siguiente apartado. Esto quiere decir que los programas que se escriben en este lenguaje no necesitan ser compilados ya que pueden ser probados en cualquier navegador lo que hace más llevadero el desarrollo de la aplicación web.

Se utiliza para dotar de dinamismo a las páginas web, es decir, para incorporar efectos, animaciones, acciones que ocurran al pulsar botones y muchas más utilidades.

Lo que hace Javascript es interactuar con el código HTML5 (aunque no es el mismo tipo de código) e incluso puede añadirse dentro de un código HTML5 siempre y cuando se etiquete y se defina ese trozo como Javascript, ya que los propios navegadores web lo pueden interpretar.

Esto se entenderá mucho mejor si comprendemos el papel que juega el DOM (Document Object Model) en las páginas web.

3.5 DOM

Es básicamente una Interfaz de Programación de Aplicaciones (API) que nos proporciona un conjunto estándar de objetos para representar documentos en HTML y XML, un modelo sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. A través del DOM, los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML, principalmente es para eso para lo que fue creado.

Esto nos permite construir elementos en un documento, navegar por su estructura, añadir, modificar o eliminar estos nuevos elementos y su contenido. Se puede acceder a cualquier parte dentro un documento HTML o XML, y se puede modificar, eliminar o añadir algo a dicha parte empleando el DOM. En algunos casos no podremos usar el DOM como por ejemplo en los subconjuntos internos y externos de HTML y XML.

Tras esta breve introducción, vamos a explicar de forma sencilla cómo funciona el DOM:

Al cargar un documento en el navegador, se crea una estructura de objetos (DOM). Ésta se puede modificar mediante el uso de Javascript, alterando los contenidos y es aspecto de la página.

Estos objetos del DOM son los que definen la forma de la ventana del navegador y todos los elementos dentro de la página. Con el uso de Javascrips podemos, a través del DOM, acceder a todos los objetos de los elementos de la página para modificar sus propiedades.

La estructura de los documentos del DOM tiene forma de árbol, aunque no está especificado que ésta sea la forma en la que deben implementarse.
Tampoco está definida la implementación de las relaciones entre objetos.

En esta imagen se muestra el esquema básico que sigue el DOM.

Figura 4 – Esquema del DOM

Unos de los nodos más utilizados es el llamado 'documento' (document), este es el nodo raíz de todos los documentos HTML y XML y es apartir de este del que se derivaran el resto de nodos.

Para representar los objetos definidos por una etiqueta se utiliza el nodo 'elemento' (element) que también es uno de los más utilizado.

También es muy importante el nodo '*Atributo*' (*Attr*) que hace posible que podamos asignar a los objetos cierta característica o valor (color, fuente...) y '*Comentario'* (*Comment*) si deseamos mostrar algo únicamente en nuestro código fuente, sin que aparezca escrito en nuestra aplicación o página web.

A parte de los tipos de nodos, disponemos de una colección de métodos o propiedades correspondientes a todos ellos, de las cuales los más útiles son las siguientes:

- *GetElementById:* Devuelve el nodo Elemento con el identificador especificado.

- *GetElementsByTagName*: Devuelve una lista ordenada de todos los nodos Elemento

que tengan como nombre de etiqueta Name.

- *CreateElement* : Crea un nodo tipo Element del tipo que se especifique.

- *GetAttribute y SetAttribute*: Devuelve o añade el valor del atributo.

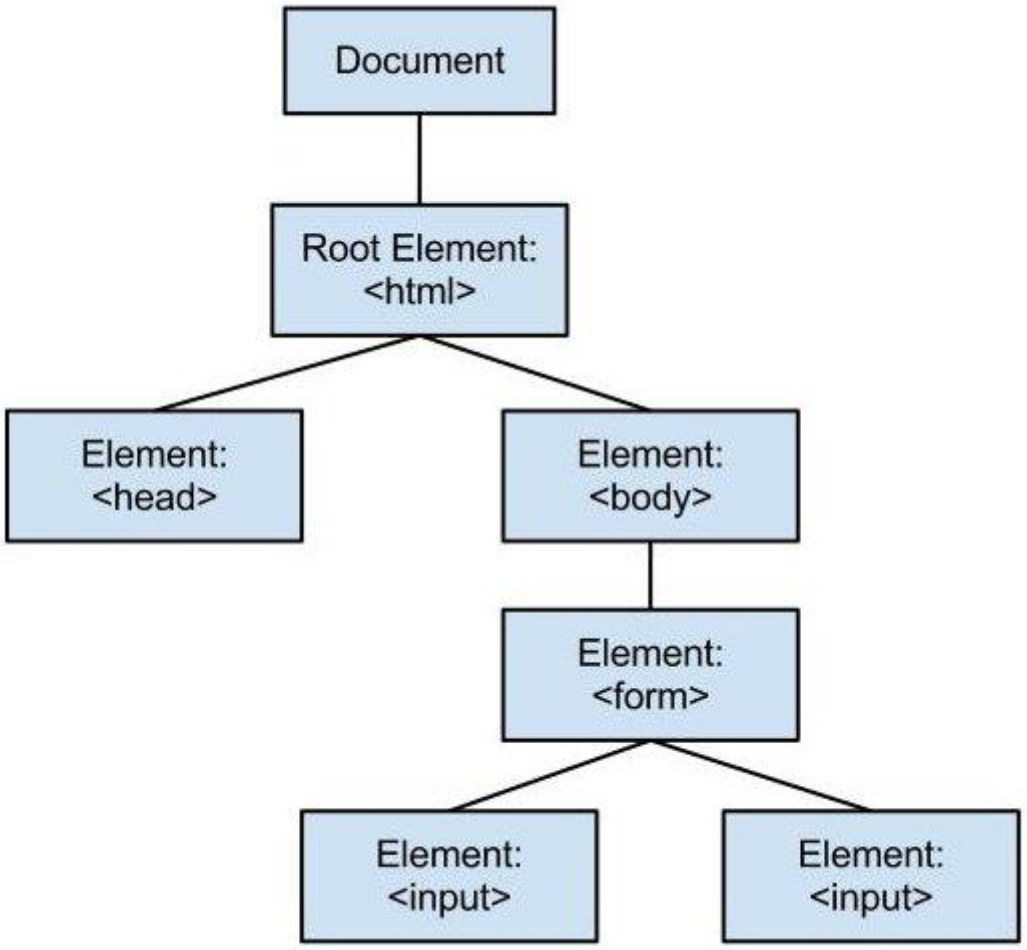


Figura 5 – Diagrama de nodos del DOM

En nuestro caso, es decir, para el lenguaje HTML, también aparecen propiedades específicas como '*images'* (documentos de imagen que aparecen en un documento), '*elements'* (conjunto de elementos de un formulario), '*value*' (valor del contenido en el formulario correspondiente), '*src'* (lugar de origen de la imagen que queremos insertar) o '*width*' y '*height'* (anchura y altura del elemento a mostrar).

Gracias a esto podremos tener un control total sobre los contenidos que aparecerán en la Web App de nuestro proyecto.

3.6 CSS3

Las hojas de estilo en cascada llamadas CSS3 son un tipo de lenguaje de programación empleado para definir la presentación general que tendrá el documento HTML de cara al cliente. Se creó con el fin de separar contenido y forma de las páginas web y a su vez tener mejor control de la apariencia de las páginas.

En cuanto a separar el contenido de la forma, este lenguaje cumplió este objetivo de manera sencilla pero de cara a tener un control sencillo de la apariencia de las páginas es más complicado e incluso actualmente algunos desarrolladores y programadores web siguen utilizando alternativas diversas y trucos para conseguir efectos visuales como el sombreado de elementos o el bordeado de estos.

Debido a estos inconvenientes ha sido necesario que este lenguaje vaya evolucionando con el paso del tiempo hasta su actual tercera versión.

Antiguamente para dar formato y color a un texto se permitía añadir atributos extra en las etiquetas de HTML que realizaran esta función, el problema de esto es que la persona que leía esta página con un navegador perdía el control sobre la forma de visualizar el texto y a la vez conllevaba el hecho de que había que definir el color tamaño y forma de cada etiqueta por separado si se pretendía que todo el tema visual de la página fuese igual, lo que llevaba mucho trabajo al escribir el código.

Por otra parte la única manera que permitía componer espacialmente era el uso de tablas. Ésta era una manera cómoda de trabajar pero conllevaba el uso de una semántica particular y que la distribución podía ajustarse bien en unos navegadores pero en otros tomaba otra forma no deseada en función de cómo decodifique cada uno de éstos el código HTML ya que no todos los navegadores tienen el mismo comportamiento.

Para entender mejor el código de CSS3 observemos el código que se muestra a continuación, en el que se ve un ejemplo de cómo crear una animación para la parte de texto que queramos.

```
<!DOCTYPE html>
<html lang="es">
< head>
< meta charset="utf-8">
<title>GIRAR</title>

< style type="text/css">
#circulo {
background-image: url(circulo.png);
height: 100px;
position: relative;
width: 100px;
-webkit-animation:gira 3s alternate infinite;
}

@-webkit-keyframes gira
{
0% {top: 0px; left: 600px;-webkit-transform:
rotate(0deg);}
50% {top: 130px; left: 200px; -webkit-transform:
rotate(180deg);}
100% {top: 400px; left:600px; -webkit-transform:
rotate(360deg);}
}
< /style>
< /head>
< body>
< div id="circulo"></div>
< /body>
< /html>
```

Figura 6 – Ejemplo de CSS3

Aunque no todo son ventajas, el uso de CSS u otro tipo de hojas de estilo también tiene sus limitaciones, a continuación analizaremos los pros y contras de éstas y veremos las mejoras más importantes que introduce la versión 3 de CSS:

Limitaciones:

- Dificultad para el alineamiento vertical; así como el centrado horizontal se hace de manera evidente en CSS2.1, el centrado vertical requiere de diferentes reglas en combinaciones no evidentes, o no estándares.

- Ausencia de expresiones de cálculo numérico para especificar valores (por ejemplo margin-left: 10% – 3em + 4px;). Aunque en la nueva versión CSS3 se propone calc() para solventar esta limitación.

- Las pseudo-clases dinámicas (como :hover) no se pueden controlar o deshabilitar desde el navegador, lo que las hace susceptibles de abuso por parte de los diseñadores en banners, o ventana emergentes.

Ventajas

- Optimización del ancho de banda de la conexión, pues pueden definirse los mismos estilos para muchos elementos con un sólo selector; o porque un mismo archivo CSS puede servir para una multitud de documentos.

- Separar el contenido de la presentación, con esto se facilita al desarrollador o incluso al usuario, la modificación de la visualización del documento sin tener que alterar el contenido del mismo, sólo modificando algunos parámetros del CSS.

- Se obtiene un control más centralizado de la presentación de un sitio web completo por lo que se agiliza mucho la actualización del mismo.

- Se mejora en la accesibilidad del documento, ya que con el uso del CSS se evitan antiguas prácticas necesarias para el control del diseño (como las tablas), y que eran un impedimento de cara a los navegadores orientados a personas con algunas limitaciones sensoriales.

Nuevas ventajas introducidas en CSS3:

- Color: La gama de colores es mucho más amplia que en versiones anteriores de las CSS, pero además, en esta tercera versión se nos permite modificar la opacidad de los elementos que queramos.

- Bordes: Pueden crearse tanto bordes en las imágenes que insertamos, como modificar su curvatura o incluso difuminarlos y crear sombras en ellos

- Fondos: Anteriormente si insertábamos una imagen de fondo, esta se repetía de borde a borde sin tener en cuenta lo que el usuario quisiese, pero esto con CSS3 ya no ocurre, ya que uno mismo indica cuántos píxeles quiere que ocupe dicha imagen. También se pueden superponer distintos fondos mediante capas.

- Texto: Podemos asignar todas las distintas fuentes que encontremos y crear sombras en las letras.

- Otras mejoras: La interfaz se puede modificar en su tamaño, se pueden crear distintos tipos de degradados, introducir columnas de texto, crear animaciones...

A continuación podemos observar un conjunto de botones con diferentes colores y formas diseñados con CSS3, en el HTML bastaría con definir el tipo de botón que es para que salga la forma como la de la imagen:

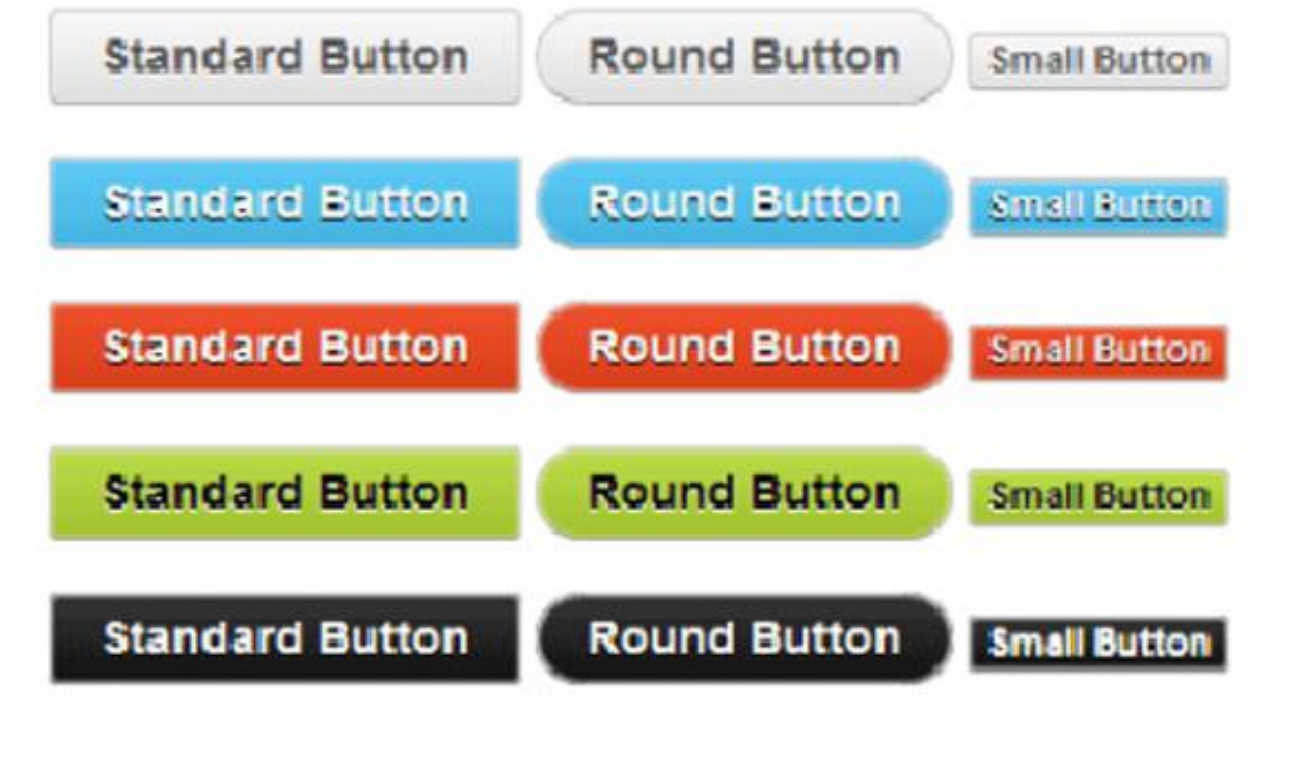


Figura 7 – Diferentes botones con CSS3

De esta manera podremos dar una bonita forma visual a nuestra aplicación web sin necesidad de cambiar el código y la forma de la página, incluso podríamos cambiar completamente este aspecto si al finalizar no nos convence el diseño.

3.7 Porqué hacer una aplicación para dispositivos móviles

Vamos a hacer básicamente un estudio del mercado y de cómo ha avanzado la tecnología en los últimos años para ver la razón por la cual hemos decidido posicionar nuestra página web orientada al mundo del dispositivo móvil.

Como todos saben los smartphones y las tablets se han impuesto en la vida de todas las personas y han ido creciendo exponencialmente en los últimos años.
Google lanzó recientemente, un estudio sobre la utilización de Internet a través de smartphones en España. El estudio, nombrado como 'Our Mobile Planet: España' muestra las principales conclusiones sobre las tendencias de uso de los consumidores móviles.

Las 5 conclusiones principales a las que llega el estudio son:

1. Los smartphones se han convertido en un elemento indispensable de nuestra vida cotidiana. En mayo de 2012 la penetración de los smartphones en España fue del 44%.

2. Los smartphones han transformado el comportamiento del consumidor. 58% de los usuarios buscan algo en sus smartphones todos los días.

3. Los smartphones ayudan a los usuarios a navegar por el mundo. 88% de los usuarios de smartphones ha buscado información local.

4. Los smartphones han cambiado la forma en que los consumidores realizan las compras. 82% han buscado un producto o servicio en su teléfono.

5. Los smartphones ayudan a los anunciantes a conectar con los consumidores. 68% ha realizado una búsqueda para móviles después de ver un anuncio.

Datos obtenidos en otro estudio revelan que en España existen más móviles que personas, debido a que mucha gente posee más de un terminal, normalmente por temas de trabajo, lo cual indica que es un sector donde hay mucha demanda y se puede sacar partido ya que se está convirtiendo en algo indispensable y que ya forma parte en la vida de todas las personas.

El empleo de los dispositivos móviles frente utilizar un ordenador para navegar por la red tiene ventajas y desventajas:

Ventajas

-La *movilidad* es la principal ventaja que tienen a favor los dispositivos móviles, ya que su facilidad para ser transportados permiten al usuario a cualquier sitio de manera que pueden realizar cualquier consulta en todo momento no importando el lugar.

-La *conectividad* es otra de las ventajas. Actualmente la facilidad que tienen estos dispositivos para conectarse a internet es muy grande, ya sea por 3G, 4G, wifi o bluetooth. Esto está relacionado con la movilidad, ya que permite conectarse desde casi cualquier parte donde se esté a internet.

-Las *diferentes funcionalidades* que incluyen actualmente estos dispositivos, como cámara de fotos, GPS o agenda, juegan a favor de ellos ya que añaden prestaciones al dispositivo haciendo todavía más práctico su empleo.

Desventajas

-Las *funcionalidades*, que hemos comentado anteriormente sin embargo no pueden compararse a las que tiene un ordenador hoy en día. Un ordenador cubre más necesidades que un dispositivo móvil.

-La *batería*, un dispositivo móvil con acceso a internet consume mucha energía por lo que su tiempo de vida es mucho menos que el de cualquier ordenador.

-El *precio* de estos dispositivos móviles suele ser caro precisamente por todas las prestaciones que incluyen actualmente y ser tan pequeños.

Dejando de lado las ventajas y desventajas de estos dispositivos, hay que darse cuenta que éste es un mercado en auge y que ofrece todavía muchas posibilidades debido a que todavía no ha llegado a su límite.

Es por esto que se ha llegado a la conclusión de que puestos a crear una página web es mucho mejor adaptarla a los móviles que hacerla sólo para el uso con una computadora.

4. ESTUDIO DE LOS DIFERENTES FRAMEWORKS

Un marco de trabajo o 'framework' es una estructura estandarizada de tecnologías con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollado. Suele incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

En el caso que nos interesa a nosotros se define un framework para desarrollar aplicaciones móviles como una estructura diseñada con bibliotecas que facilita la creación de una webapp. Es algo así como una plantilla vacía de una web a la cual el desarrollador le va añadiendo los elementos a su gusto y la configura a sus necesidades.

La ventaja que tiene el uso de estos frameworks es principalmente la rapidez en el desarrollo de las aplicaciones, ya que el programador no tiene que escribir todo el código ya que existe una capa ya creada y el programador únicamente la modifica y/o la utiliza a su gusto. De esta manera se reutilizan muchos componentes software.

El framework más conocido y utilizado para lo que se propone hacer se llama jQuery, que se ha ido desarrollando y mejorando desde 2005.

El principal problema es que sigue siendo JavaScript, con la necesidad añadida del aprendizaje de un nuevo entorno y una nueva forma de trabajar.

El segundo problema es que el framework nos obliga a descargar una librería más desde nuestro sitio Web o alguno de los servidores asociados lo que puede aumentar el tiempo de carga de nuestras páginas. La librería de interface de usuario puede llegar a ser demasiado pesada aunque no tenemos que incluirla completa (e incluso podemos no utilizarla en absoluto).

El tercer problema es que es muy lento.

Es por esto que se decidió no utilizar jQuery e investigar nuevos frameworks más recientes que todavía no están tan desarrollados, con el fin de aprender mejor el uso de un framework y a su vez intentar buscar una alternativa a jQuery que tenga la misma funcionalidad o mejor.

Para nuestro proyecto hemos decidido estudiar 3 frameworks diferentes de los cuales elegiremos uno para la realización de nuestro proyecto. Éstos son:

4.1 Sencha Touch

Este framework que hemos estudiado se basa en librerías JavaScript y fue creado por la empresa Sencha, resultante de la unión de de ExtJs, jQTouch y Raphaël.

Sencha Touch es un framework que permite desarrollar aplicaciones web para dispositivos móviles táctiles, dando prácticamente la sensación de que son aplicaciones nativas de los sistemas operativos de los dispositivos.

Fue el primer framework construido a partir de estándares web, y diseñado para aprovechar específicamente la potencia, flexibilidad y optimización del lenguaje HTML5, CSS y Javascript. Como ya sabemos, la utilización de HTML5 le ha dotado la posibilidad de ofrecer componentes como el audio o el video, además de un proxy local para almacenar información incluso en estado "*offline"* (sin conectar a la red).

Respecto a su hoja de estilos (CSS), este framework ha implementado una capa de estilos independiente de la resolución que tengan los dispositivos móviles. Gracias a una combinación de tamaños CSS3, ha logrado adaptar los componentes de una interfaz de usuarios a la resolución de cada dispositivo (deben ser compatibles, por supuesto). Además, se aprovecha de una tecnología diferente de los otros frameworks (SASS) que se basa en implementar una capa de CSS en la que se añaden variables y funciones, con lo que se puede cambiar la presentación de una aplicación totalmente.
Sencha proporciona una librería con múltiples widgets como pestañas, formularios, listas... Todo ello pudiendo utilizar temas creados por el usuario.

En cuanto a la manipulación del DOM, hay que destacar el gran tamaño de su librería, con varias colecciones, paquetes y clases para facilitar la reutilización de código.

Todo se basa en heredar y codificar nuestros propios objetos, diseños y componentes con códigos que ya existen. A continuación, hacemos referencia a las clases principales del Framework siguiendo su jerarquía de herencia o el árbol DOM.

Mediante el siguiente script se consigue llamar al DOM:

```
<script type="text/javascript">

Ext.onReady(function() { //se usa para inicializar los componentes en el momento adecuado
  Ext.BLANK_IMAGE_URL = "common/ext/resources/images/default/s.gif";
  Ext.QuickTips.init();

});

</script>
```

'*Ext.onReady'* es la función más importante, sirve para inicializar los componentes justo cuando el DOM ya ha sido cargado, lo cual significa que cuando el DOM esté listo ejecutará la acción. Además de ésta, otra función interesante es '*renderTo'*, que recibe un elemento DOM para renderizar el componente dentro de este elemento

Sencha Touch t presenta grandes características para decantarnos por este framework a la hora de realizar aplicaciones dinámicas. Cabe destacar, que presenta un conjunto de datos muy potente y robusto, que permite solicitar y obtener datos a a través de diferentes fuentes como *"Ajax"*. Además permite unir esos datos a plantillas o listas HTML.

A continuación vamos a mostrar las ventajas e inconvenientes de este framework a la hora de realizar Web Apps.

Ventajas

- Nos permite crear aplicaciones complejas utilizando componentes predefinidos.

- Evita el problema de tener que validar el código para que funcione bien en cada uno de los navegadores (Firefox, IE, Safari, Opera etc.) al estar estandarizado.

- Una ventaja fácil de apreciar es el funcionamiento de sus ventanas flotantes, en este aspecto es superior a cualquier otro framework.

- Relación entre Cliente-Servidor balanceado: Se distribuye la carga de procesamiento entre ellos, permitiendo que el servidor pueda atender más clientes al mismo tiempo, es decir, que se repartan el uso de recursos entre esas dos partes.

- Eficiencia de la red: Disminuye el tráfico en la red al contar las aplicaciones con la posibilidad de elegir qué datos desea transmitir al servidor y viceversa.

- Comunicación asíncrona. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, es decir, se puede cargar información sin que el cliente se cuenta.

- Al ser "herencia" de ExtJs, este framework estaba desarrollado en su mayor parte, por lo que no se ha requerido de demasiados cambios para hacerlo funcional. Esto evita muchos fallos implícitos en el inicio de un nuevo proyecto.

- Hay una gran cantidad de componentes muy diversos, por lo que podemos crear el diseño de una Web App de muchas formas diferentes con un código similar.

- El que haya una empresa como Sencha detrás de este framework, hace que su su desarrollo sea mucho más sencillo.

- Presenta un Soporte para geolocalización y mapas, incluyendo un componente que implementa el API de Google Maps.

Inconvenientes

- Al ser el primer framework que apareció, los errores o bugs presentes fueron corregidos sin poder apoyarse en otros.

- Es uno de los framewoks más pesado, es decir, que los recursos de sus aplicaciones ocupan más que los de la competencia, detalle por el cual, muchos usuarios pueden renegar de Sencha Touch y decantarse por otros.

- Su principal inconveniente es que el mayor peso de las aplicaciones que sean lentas y se produzcan sobrecargas al usarlas.

Otra de las grandes desventajas de este framework es su compatibilidad con los dispositivos móviles. Se ha creado la segunda versión de Sencha Touch, pero aunque tiene muchas variantes y mejoras que su primera versión, todavía está desarrollada en HTML5 para muy pocos sistemas operativos de móviles. Únicamente es compatible con Iphone (todos sus productos), Android (versiones desde 2.3 hacia adelante) y algunos dispositivos de BlackBerry (un grupo muy reducido). Esto hace que esté un nivel por debajo de LungoJS.

Las aplicaciones en Sencha Touch funcionan mejor cuando siguen estructuras simples que este mismo framework proporciona para que el código sea sencillo y entendible. El primer paso es establecer la estructura. Inicialmente lo único que se necesitan son dos archivos y copiar la carpeta de Sencha Touch. Los archivos son:

app.js : Archivo donde se definen las configuraciones de tu aplicación.
Touch : La copia de la carpeta descargada de Sencha Touch.
index.html : una página inicial de HTML que incluye el archivo principal de Sencha Touch.

Vamos a ver un index muy básico de Sencha Touch

```
<!DOCTYPE html>
<html>
<head>
    <title>Iniciando con el desarrollo en Sencha Touch</title>
    <link rel="stylesheet" href="touch/resources/css/sencha-touch.css" type="text/css">
    <script type="text/javascript" src="touch/sencha-touch-all.js"></script>
    <script type="text/javascript" src="app.js"></script>
</head>
<body>
</html>
```

Tras esto, creamos las interfaces gracias al archivo *app.js* comentado anteriormente. Aquí observamos la función *TabPanel*, que crea una interfaz de paneles agrupados en tabs (en este caso mostramos sólo con uno) con el siguiente código:

```
Ext.application({
    name: 'Sencha',

    launch: function() {
        Ext.create("Ext.TabPanel", {
            fullscreen: true,
            items: [
                {
                    title: 'Home',
                    iconCls: 'home',
                    html: 'Welcome'
                }
            ]
        });
    }
});
```

Figura 8 - Javascript para Sencha Touch

El resultado de combinar estos dos códigos anteriores tendría ya una forma básica de aplicación web, aunque estuviese vacía, podría observarse ya un título y el aspecto visual que tendría.

4.2 LungoJs

LunjoJS es un framework de desarrollo para aplicaciones móviles web, que sigue el estándar HTML5/CSS3 para la creación del código fuente. Está desarrollado por *TapQuo*, que es una empresa española afincada en Bilbao, especializada en desarrollo web y sobre todo en tecnologías Javascript.

El desarrollo del framework estuvo motivado cuando Javier Villar Jimenez se puso a investigar sobre tecnologías de desarrollo para dispositivos móviles, una vez consiguió su primer iPhone (que no llegó a venderse en España). Al desarrollar primeramente en nativo (Objective-C) y darse cuenta de que en este caso su desarrollo no sería compatible con distintos dispositivos (para Android debería usar su API Java, para Microsoft sus propias tecnologías...), decidió volver a sus raíces y probar el desarrollo de webapps. En ese momento empezaban a sonar las tecnologías HTML5 y CSS3, y al disponerse de un estándar de futuro para estos desarrollos se abrieron las opciones.

Al investigar los dos frameworks dominadores en el desarrollo de webapps de estos momentos (*Sencha y jQuery Mobile*), se encontró con algunas dificultades e inconvenientes que le hicieron plantearse la implementación de su propio framework.

Estos inconvenientes son derivados de la filosofía que han adoptado los dos grandes desarrollos, al adaptar sus bibliotecas de desarrollo para aplicaciones de escritorio al entorno móvil. Esto hace que las bibliotecas sean relativamente pesadas al utilizar mucho código que en realidad no se necesita en el mundo móvil. Además los eventos táctiles de estas pantallas son distintos a los eventos de ratón que se utilizan en los entornos de escritorio, y no siempre están bien resueltos.

Otro de los aspectos que quería conseguir es tener un entorno realmente HTML5 /CSS3 con el que trabajar.

La filosofía de Lungo es totalmente distinta: el framework implementará las características que estén descritas en el estándar HTML5, y por lo tanto sólo dará soporte a aquellos dispositivos que tengan una compatibilidad real con este estándar.

Presumiblemente, cada vez los sistemas van a tener una compatibilidad con HTML5 más depurada, por lo que a priori parece una buena estrategia de crecimiento.

La licencia de este framework es *GPL v3* para desarrollo de aplicaciones (es decir, software libre). Si se quiere usar de forma empresarial, se debe contratar una licencia comercial de pago.

A la hora de comenzar con un desarrollo en este framework, en los tutoriales y en la misma descarga de las bibliotecas, se nos recomienda usar una serie de ficheros javascript para organizar el desarrollo. Por supuesto, se puede usar otra organización si lo creemos necesario, pero la propuesta es realmente cómoda de usar y efectiva:

app.js: Para inicializar la aplicación.

data.js: Para las llamadas al sistema de almacenamiento webSQL.

events.js: Para los manejadores de eventos.

services.js: Para las llamadas a servicios remotos (ejemplo, llamadas AJAX).

view.js: Para la definición de vistas, mediante plantillas (templates).

A continuación mostramos las ventajas e inconvenientes que hemos encontrado en este framework a la hora de realizar Web Apps.

Ventajas

- Creación sencilla de aplicaciones web para dispositivos iOS (Apple), Android y Blackberry

- Desarrollo de aplicación HTML5 con estructura semántica en todo el proyecto, comenzando por la plantilla en HTML, apoyándose en los estilos CSS y terminando con su API Javascript.

- No necesita líneas de código en Javascript para realizar prototipos de aplicaciones, solamente con el HTML se puede mostrar el resultado en el navegador o dispositivo móvil.

- Utiliza una librería para el manejo del DOM de la página extremadamente ligera, llamada QuoJS y desarrollada también por TapQuo, de manera similar a como trabaja jQuery.

- Aprovecha las capacidades de los móviles actuales.

- Implementación sencilla de características HTML5 como WebSQL, orientación, conexión...

- Captura eventos táctiles como swipe, tap, doble tap...

- Puede extenderse la funcionalidad del frameworks mediante plug-ins o complementos, que en este entorno se llaman Sugars.

- Diseño totalmente personalizable.

- Permite distribuir las aplicaciones tanto en sitios web como en las distintas stores (Google Play, App Market de Apple...).

Inconvenientes

- A pesar de ser muy parecido a JQueryMobile, la cantidad de Web Apps, plugins … disponibles, así como la participación de desarrolladores en foros, es infinitamente menor (está poco extendido).

- La ayuda que proporciona LungoJs y sus colaboradores es muy poca, lo que dificulta la posibilidad de solventar problemas y errores que surjan a lo largo del desarrollo.

- Su funcionamiento está limitado al navegador *GoogleChrome*, tanto en *Internet Explorer* como en *Mozilla Firefox* no ofrece un buen comportamiento.

Como ya se ha comentado anteriormente las Web Apps creadas con "LungoJS solo son soportadas en dispositivos que den soporte real a HTML5/CSS3/JavaScript.

Por lo tanto, a día de hoy, su compatibilidad está asegurada con los sistemas más utilizados (*Blackberry, Android e iOS*) y *Google Chrome* como navegador de escritorio.

Desde septiembre de 2012 (inicio del PFC) a enero de 2013 ha habido una constante mejora en Lungo, pasando por las versiones: *1.1.1, 1.1.2*, 1.2 (usada en este PFC) y la recientemente publicada *2.0.*
En este tiempo se ha acentuado el desarrollo de los sistemas operativos compatibles con Lungo, mientras que el resto de sistemas operativos se han quedado obsoletos.

No cabe duda que la visión de futuro de LungoJS es increíble y el hecho de estar desarrollado con muy pocos medios y apenas un par de personas es algo muy meritorio.

Veamos un ejemplo del funcionamiento de este framework:

En cuanto a la definición del marcado en la página HTML, se utiliza la nomenclatura recomendada para las páginas en HTML5, es decir, se definen como secciones que en su interior contendrán cabeceras, pies de página y artículos.

Una sección puede contener varios artículos. Si pensamos en la organización que pueda tener un blog, las secciones de la página pueden ser por ejemplo una lista con el archivo por meses de las entradas, y otra sería el sitio en el que se indica el texto de las entradas. Los artículos serían cada una de las entradas del blog.

En LungoJS, definiríamos la sección principal mediante el código indicado a continuación:

```
<section id="main">
    <header data-back="home blue" data-title="Título de la sección"></header>

    <footer class="toolbar"></footer>

    <!-- content -->
</section>
```

Figura 9 – Código de una seccion

En esta sección se definen además la cabecera, en la que se indica el botón de atrás, que al estar marcado semánticamente, se define con el icono de *HOME* y de color azul, mediante el atributo *data-back*. También se indica el título de la sección, mediante el atributo *data-title*.

Para el *footer*, se le indica en la clase que se va a comportar como una barra de herramientas, por lo que estaría preparada para que indicáramos botones en su interior.

Además de cabeceras y pies de sección, en su interior se pueden indicar artículos, tal y como se muestra en el ejemplo siguiente:

```
<section id="main">
    <!-- header -->

    <!-- footer -->

    <article id="first_article">
        <!-- content -->
    </article>

    <article id="second_article">
        <!-- content -->
    </article>

</section>
```

Figura 10 – Esquema del contenido de la sección

En la imagen siguiente se puede observar el resultado que tendría en el navegador:

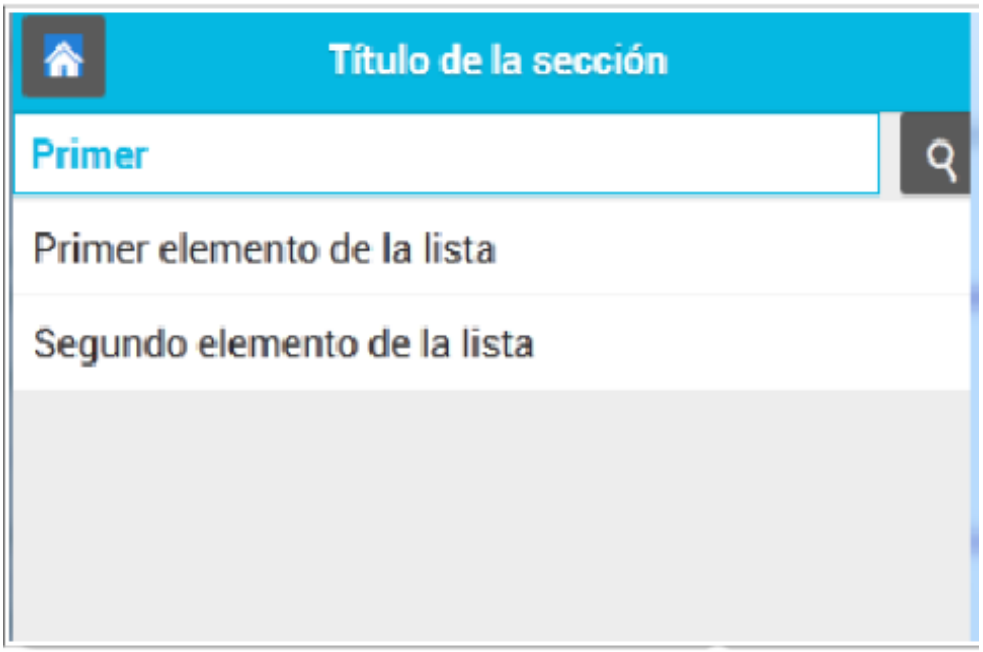


Figura 11 – Aspecto visual en el navegador

4.3 EnyoJs

Este framework tiene como objetivo crear aplicaciones únicamente funcionales bajo el sistema operativo creado por *HP* denominado *"webOS*".

Lo primero que se va a hacer es explicar qué es webOS y cómo surgió EnyoJS.
Lanzado por primera vez en 2009 con la Palm, un aparato predecesor del teléfono inteligente, webOS recibió buenas críticas por su intuitiva interfaz de usuario y sus herramientas para hacer múltiples tareas a la vez. Entre ellas una función que permitía a los usuarios ver las aplicaciones abiertas como una pila de cartas en la pantalla que se podían cerrar deslizando el dedo sobre ellas o poner en primer plano dándoles un toquecito. A pesar de las críticas favorables de los expertos, el software nunca ganó demasiado terreno entre los consumidores, ni siquiera después de que Hewlett Packard adquiriese una moribunda Palm por unos 1.380 millones de euros en 2010 e intentara resucitar sus productos móviles.

HP cerró su negocio de aparatos móviles, que incluía webOS, a finales de 2011. Pero en septiembre de este año la compañía lanzó Open webOS, su objetivo es poner a disposición de desarrolladores y fabricantes bajo licencia Open Source tanto el sistema operativo como el entorno de desarrollo del mismo, conocido bajo el nombre de **ENYO**.

El primer paso dado por Phoenix ha sido conseguir que webOS funcione como aplicación en teléfonos inteligentes ya existentes que operan con Android.

El equipo de Phoenix ha construido una aplicación que sí funciona (aunque muy lentamente) en un teléfono inteligente Nexus S, y Zakutny afirma que esperan tener disponible una versión más rápida de la aplicación la tienda Google Play dentro de unos dos meses (aunque no especifica cuánta de la funcionalidad del webOS original proporcionará la aplicación). El grupo quizá la ofrezca gratis con la esperanza de que hará que más gente se interese por webOS.
Cualquier aparato nuevo operando con webOS necesitará una nueva cosecha de aplicaciones. Phoenix espera resolver este tema usando OpenMobile, cuya tecnología

hace que las aplicaciones Android puedan funcionar en aparatos que no tienen Android como sistema operativo. Al igual que webOS, Open webOS incorpora estándares de programación Web, cuya intención es facilitar a los desarrolladores Web crear aplicaciones o portarlas al sistema operativo.

Figura 11 - Interfaz del Sistema Operativo webOS

EnyoJS trabaja con una biblioteca llamada Onyx, es una librería bastante completa, he aquí unos ejemplos:

Figura 12- Ejemplos de botones disponibles en la librería Onyx

Ventajas

Todas las ventajas que aparecen a continuación están referidas a la versión 2.0 de libre código.
Al comenzar este proyecto de final de carrera esta versión no estaba disponible, motivo que provocó que no empleáramos este framework.

- Libre y de código abierto: 100% libre de utilizar, Enyo está disponible bajo la licencia Apache, versión 2.0.

- Extensible: con un pequeño y sólido núcleo, Enyo es modular y está diseñado para ser ampliado por la comunidad de desarrolladores.

- Construido para manejar la complejidad: modelo de componentes elegante Enyo hace que sea fácil de construir y mantener incluso las aplicaciones más complejas.

- Optimizado para móvil: enyo tiene sus raíces en móvil y fue construido desde cero para brillar en las tabletas y los teléfonos.

- Ligero y Rápido: enyo es pequeño (núcleo es <25k gzip) y ajustado para la velocidad y capacidad de respuesta en todas las plataformas soportadas.

Inconvenientes

- Aunque en la versión actual de enyo se ha intentado que éste sea soportado por la mayoría de plataformas móviles y de escritorio, todavía queda un largo camino que recorrer.

- La información de Enyo se encuentra exclusivamente en inglés y al comienzo del PFC no había ninguna Web App que usara esta tecnología como ejemplo.

- Este framework está dirigido principalmente a webOS, un sistema operativo que como se ha comentado anteriormente está obsoleto a día de hoy.

A pesar de la reciente liberación de código para hacer que este framework construya aplicaciones funcionales en todas las plataformas, al inicio del proyecto esto no estaba asegurado, ni siquiera estaba disponible la versión actual.

En ese momento este framework se empleaba en la elaboración de Web Apps destinadas al sistema operativo webOS, excluyendo el resto de sistemas operativos, entre los cuales se encuentran los más extendidos en el mercado como Android, i0S y Windows Phone.

Esto supuso una gran desventaja frente al resto de frameworks.

4.4 Elección de un framework

Una vez vistos los tres frameworks que se han investigado como alternativa a JQuery Mobile se ha tenido que decidir cuál de éstos era el más conveniente.

El primero en ser descartado ha sido EnyoJs porque como se explicaba anteriormente cuando este proyecto de fin de carrera dio comienzo éste framework no estaba disponible para los usuarios libres ya que había que pagar una licencia de uso y sólo servía para plataformas con sistema operativo webOs, el cual está casi obsoleto ya que casi ningún dispositivo utiliza esto como se ha explicado anteriormente.

Tras descartar EnyoJs, quedan LungoJS y Sencha Touch como frameworks para nuestra aplicación web. Tras analizar detalladamente las características y funcionalidades de ambos se ha llegado a la conclusión que ambos cubren nuestras necesidades ya que son muy similares. Pero cabe destacar que la programación con Sencha Touch es más complicada ya que su lenguaje Javascript es muy extenso y complejo mientras que con LungoJs es mucho más intuitivo.

Una vez dicho todo esto, lo siguiente que se ha tenido en cuenta es el peso de los archivos, ya que las bibliotecas de Sencha Touch son tan extensas que ocupan mucho espacio lo que implica que la carga de sus páginas sean más lentas y pesadas, mientras que LungoJs está más comprimido por lo tanto sus acciones se realizarán con más rapidez y consumen menos ancho de banda.

Por tanto nos hemos decantado por utilizar LungoJs que aunque todavía está en desarrollo se cree que va a ser lo más conveniente debido a su esquema intuitivo y sencillo así como a que trabaja a más velocidad que Sencha Touch, además de que al haber sido creado por un equipo de Bilbao incluye tutoriales y foros en castellano, lo que es otro punto a favor en nuestra opinión.

5. TECNOLOGÍAS EMPLEADAS

Una vez explicados todos los conceptos básicos necesarios para entender el proyecto y elegido el marco de trabajo necesario que se adecua a nuestro propósito nos centraremos en las diferentes tecnologías y programas que hemos utilizado y el papel que han desarrollado en la realización del proyecto, algunas de las cuales ya hemos comentado anteriormente:

-LungoJs
-HTML 5
-CSS3
-Javascript
-PHP
-Ajax
-Sublime Text
-Filezilla
-OpenData Navarra
-Google Maps API
-Google Chrome + Chrome Inspector
-Lungo Sugar
-Phpmailer

A continuación explicaremos brevemente la utilidad de cada una de ellas y a lo largo del desarrollo se irá viendo para que se hayan empleado.

En el apartado anterior ya hemos justificado porque hemos elegido LungoJs como framework para realizar nuestra webapp.

Como también se ha explicado anteriormente, en toda aplicación web es imprescindible el uso de HTML5 y CSS3 para montar el prototipo o 'esqueleto' de la página así como todo lo relacionado los elementos visuales. La utilización del lenguaje CSS3 para definir la forma, color y tamaño de los elementos de la página además nos facilita mucho la accesibilidad ya que la webapp se carga más rápido debido a que ya tenemos definidos los estilos visuales para todos los documentos y elementos que se carguen.

También hemos explicado el uso de Javascript añadir dinamismo a la aplicación ya que es el lenguaje de programación más utilizado en este proyecto.

Aunque no hemos explicado lo que es PHP ya que no lo hemos usado mucho en el proyecto, sí que es importante conocer su utilidad. Éste es un lenguaje que se ejecuta en el lado del servidor por lo que el cliente no ve lo que hace, tan sólo ve el resultado final. Al ejecutarse en el lado servidor permite conectarse a bases de datos, realizar consultas, enviar correos, subir archivos y muchas utilidades más y además es compatible con todos los navegadores y se puede incorporar en cualquier archivo HTML.

Similar a PHP, AJAX (JavaScript asíncrono y XML) es una técnica de desarrollo web para crear aplicaciones interactivas que se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. Sirve para hacer modificaciones en la página sin necesidad de recargarla.

Para la realización de la aplicación web se ha utilizado el programa Sublime Text v2.0 que permite la escritura de código en todos los diferentes lenguajes de programación y es tan fácil de usar como un editor de texto, además de ser gratuito.

Filezilla es el programa que hemos utilizado para poder volcar todos los archivos al servidor web donde tenemos almacenada la webapp.

De OpenData Navarra y Google Maps hablaremos más detalladamente en el desarrollo de la aplicación ya que tienen mucho peso en este proyecto y necesitan ser explicados más detalladamente.
El navegador web Google Chrome tiene añadida una herramienta llamada Chrome Inspector (click derecho en cualquier página web) que permite ver el comportamiento interno de cualquier web así como sus errores de compilación y código. Esta herramienta también la hemos utilizado mucho para ver los fallos de nuestra webapp así como si iba realizando las acciones que se le iban pidiendo.

Lungo Sugar no es más que un conjunto de plugins adicionales que existen para añadir a LungoJs, son archivos Javascript y CSS ya diseñados que permiten añadir complementos a nuestra aplicación como un calendario, una galería de fotos o simplemente notificaciones interactivas y animadas.

Finalmente nos hemos ayudado de otro complemento llamado PhpMailer que sirve para permitir a un archivo PHP enviar correos electrónicos desde el servidor en el que se encuentra.

6. DESARROLLO DE LA APLICACIÓN

6.1 Esquema de las páginas

Lo primero que se ha planteado ha sido el esquema que tendrá la aplicación web, es decir, el diagrama de navegación que seguirán y las páginas que lo formarán.

La idea es hacer una página de inicio principal a partir de la cual se pueda dirigir el usuario a distintas secciones para poder encontrar el centro que desea.

En esta página de inicio se dispondrán cuatro botones:

Uno que servirá para acceder a una lista de todos los centros disponibles en toda Navarra y a parir de esta lista acceder a cada uno de ellos y poder ver información sobre éste, un mapa con su posición, una galería de imágenes y enviar un correo.

El segundo botón servirá para acceder a la sección de mapas, aparecerá la posición actual del usuario de la aplicación y dispondrá de un botón para buscar el centro más cercano a dicha posición.

El tercer botón será para facilitar la búsqueda de los centros filtrándolo por localidades ya que si el usuario ya sabe dónde va a estar el centro no es necesario que lo tenga que buscar en una lista entre 368 centros, de esta manera se hace más sencillo el acceder al centro deseado.

El cuarto botón es similar al anterior, también sirve para filtrar los resultados pero en lugar de por localidades por tipo de centro, ya que si el usuario necesita un hospital o un consultorio no es necesario que lo tenga que buscar entre todos los tipos ya que son demasiados.

La mejor manera de comprender el esquema de navegación que tendrá nuestra aplicación web es observar el diagrama que vamos a mostrar en la siguiente página.

Figura 13 – Diagrama de navegación de la webapp

En todo momento el usuario tiene disponible un botón para volver a la sección anterior o a la página de inicio.

Además se observa que como mucho tiene que recorrer 3 secciones, es decir, la navegación es lineal y apenas tiene que hacer 3 clicks para llegar a donde desea el usuario lo que hace que sea una navegación sencilla e intuitiva.

6.2 Conexión a los datos

Lógicamente lo primero que necesitamos nuestra webapp son los datos con los que vamos a trabajar.

Estos datos son los extraídos de OpenData Navarra y están almacenados en una base de datos MySql, un software libre para almacenar datos en la nube que tiene un lenguaje de programación propio llamado SQL. Esta base de datos puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

A estas bases de datos hay muchas maneras de entrar y modificarlas para trabajar con ellas, nosotros hemos usado un programa online llamado 'PhpMyAdmin' el cual permite acceder a la base de datos MySql y exportar, modificar e importar datos en ella.

Figura 14 - Base de datos administrada desde phpMyAdmin

Para conectarse a una base de datos MySql es necesario el uso de PHP ya que este se ejecuta en el lado del servidor, es decir, no puedes acceder a un servidor con HTML ya que este está en el lado cliente mientras que la base de datos está en el lado servidor, por lo tanto lo que hacemos es que con el script Services.js se llame a un php que se ejecute en el lado del servidor y será este el encargado de conectarse a la base de datos y devolvernos todos los datos en una tabla de valores. De esta manera además el usuario no notará ningún cambio ya que los archivos PHP no se muestran en pantalla ya que como hemos comentado se ejecutan internamente en el propio servidor, lo que favorece la estética de la aplicación ya que no cambia de apariencia ni se recarga la página.

La manera que tiene LungoJs de trabajar de esta manera es utilizando AJAX, mediante un comando propio implementado en Lungo llamado *json* que funciona realizando una petición al archivo php y devolviendo el resultado en un formato llamado json.
Este es el código javascript del services.js que nos permite utilizar el php:

```
$$.json('mysql2.php',
{},
function(data) {
App.Data.cacherest(data);
});
```

A su vez el PHP llamado ‘mysql2.php’ recibe esta petición y se conecta a la base de datos que piden, utilizando el usuario y contraseña almacenados en el propio PHP, este archivo también contiene unas órdenes específicas en lenguaje SQL para una vez conectado a la base de datos seleccionar la tabla específica y los datos a recoger. Finalmente devuelve estos datos a Lungo en forma de objeto JSON.

Por defecto HTML5 trae acceso a una base de datos interna del propio código en este caso de SQL que se guarda en la caché de la página web con la que se puede trabajar como con cualquier base de datos. Esto nos permite una vez que Lungo ha recibido los datos del servidor los almacene en esta base de datos interna.

Con esto conseguimos evitar que cada vez que el usuario haga una consulta a los datos tengamos que conectarnos con el servidor externo, acceder a él, consultar los datos y volver a la página, todo esto consumiría mucho tiempo y evitaríamos que el usuario tuviera que hacer una comunicación externa 3G lo que consume mucha batería.

De esta manera haremos que solo se conecte a la base de datos externa la primera vez que el usuario acceda a la página y lo almacene en la base de datos local de HTML5 y el resto de consultas se hagan en esa base de datos.

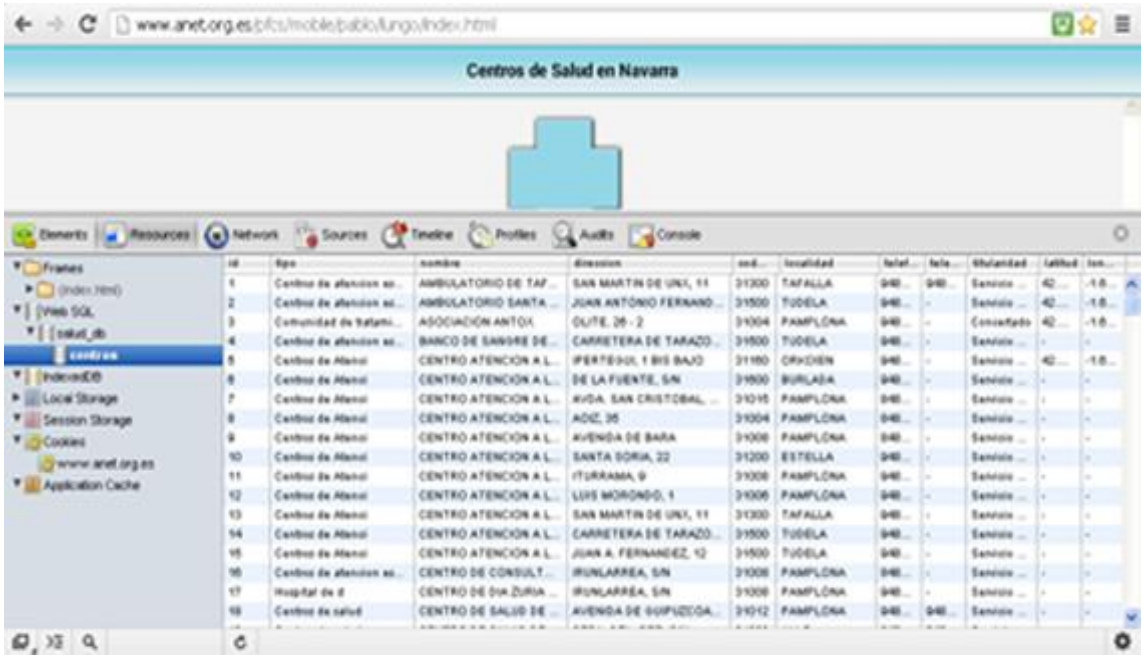


Figura 15 – Base de datos SQL interna

En esta imagen podemos observar accediendo a Chrome Inspector (botón derecho en Google Chrome y luego pinchar en Inspeccionar elemento) como en nuestra página se ha creado una base de datos SQL interna con todos los datos.

Aquí mostramos un esquema de cómo sería todo el proceso de petición, conexión y selección de la base de datos mediante el PHP:

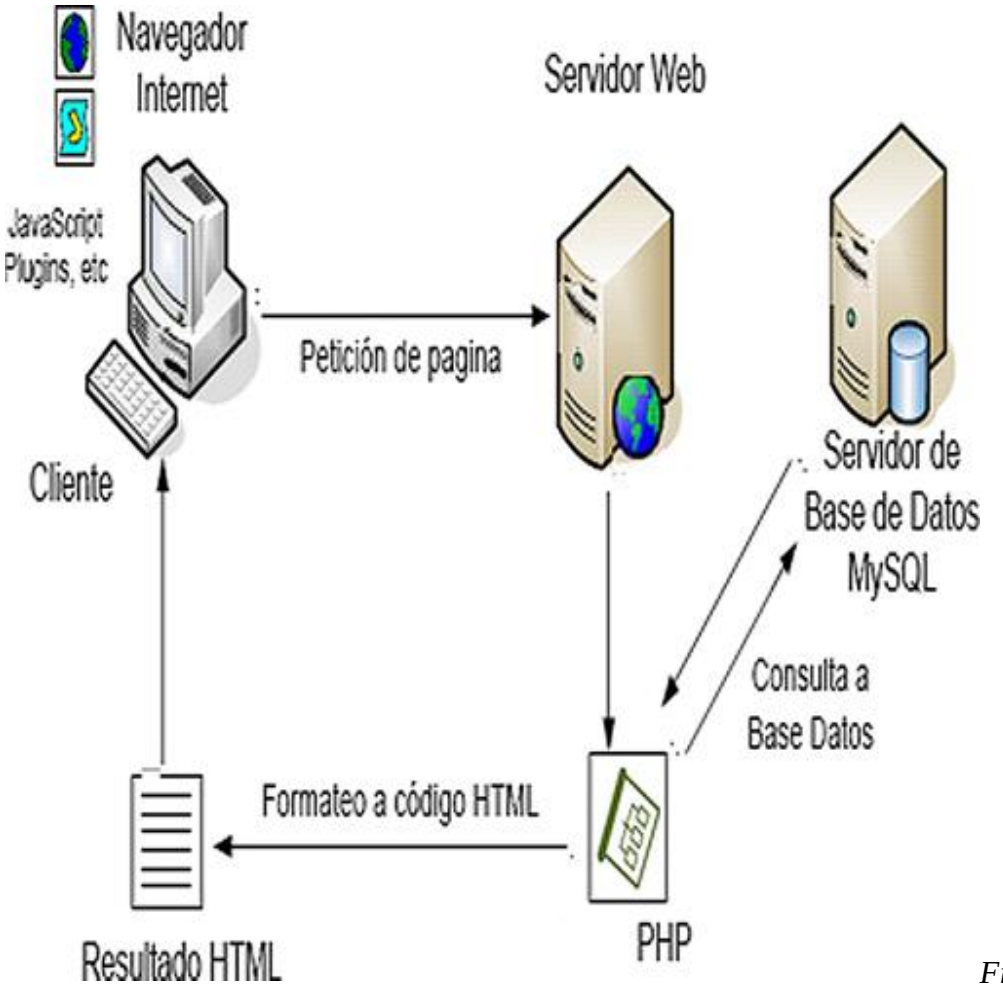


Figura 16

6.3 Aspecto visual

El diseño de la webapp se realiza mediante HTML5 para crear la forma que tendrán las páginas, ordenando sus diferentes secciones y contenidos de manera ordenada.
Es aspecto visual, es decir, los colores de los botones, encabezados, tipos de letra y demás, se modifica utilizando las hojas de estilo en lenguaje CSS.

Existen algunas hojas de estilo predefinidas y ya creadas con diferentes temas de formas y colores, en nuestro caso hemos escogido la que incluye Lungo por defecto llamada 'lungo.theme.default2.css'. En cuanto a las formas y tipos de letra no hemos cambiado nada debido a que las que venían incluidas son las más comunes y se adecúan a nuestros requisitos.

Lo que sí hemos modificado son los colores, definiendo para la cabecera un degradado azul claro que será el color básico que deseamos como tema de nuestra aplicación web:

```
@import "lungo.theme.default.font.css";

.app {
  background: #ffffff;
  font-family: 'Roboto', Helvetica, Arial, sans-serif;
}
/* @group <header> & <footer> & <article> */
header {
  background: #93d6e7 -webkit-gradient(linear, left top, left bottom, color-stop(0.25, #93d6e7),
  border-top: 1px solid #05bde9;
  border-bottom: 1px solid #07acd0;
  box-shadow: 0 4px 0 rgba(0, 0, 0, 0.1);
}
footer {
  background: #ffffff -webkit-gradient(linear, left top, left bottom, color-stop(0.25, #2c2c2d),
  border-top: 1px inset #1c1c1c;
}
.title {
  color: #000000;
  text-shadow: 0px 1px 0px rgba(0, 0, 0, 0.2);
}
article {
  background-color: #f4f4f4;
}
article .title {
  color: #ffffff;
  text-shadow: 0px 1px 0px #fff;
}
```

Figura 17 - 'lungo.theme.default2.css' modificado

6.4 Navegación

La navegación a través de nuestra webapp se puede dividir en dos secciones que juntas realizaran las acciones necesarias para que podamos navegar dinámicamente por nuestra webapp.

Navegación estática:

Ya hemos visto anteriormente como va a ser el esquema o diagrama de nuestra aplicación web y sus distintas secciones.

Denominamos navegación estática a viajar entre distintas secciones, estas secciones son fijas y están definidas en el HTML principal de manera que la navegación entre ellas también está ya predefinida. Es decir, si pinchamos en el botón que vaya a la sección de búsqueda, este botón ya tiene indicada la función mediante la cual la página debe viajar a otra sección. Esto es muy sencillo con HTML 5, basta con indicar que es un enlace a otra sección y poner el nombre de la sección:

<a href="#seccion a la que queremos que se dirija" data-target="section" class="button">

La cuestión es que queremos que sea una aplicación dinámica, es decir, si seleccionas un centro de salud concreto, en la sección correspondiente queremos que salgan los datos de dicho centro, si seleccionas otro distinto, que en la misma sección cambien los datos por este otro. Entonces lo que hacemos es que las secciones sean plantillas vacías que contengan solo el esquema, colores y formas en el que se presentarán los datos. En resumen, con HTML5 realizamos la navegación entre diferentes plantillas o contendedores que se rellenaran en función del elemento seleccionado por el usuario.

Navegación dinámica

En nuestro caso denominamos de esta manera a la parte destinada a que los contenidos de nuestra aplicación vayan variando y mostrándose en función de las elecciones del usuario.

Los códigos javascript de LungoJs (events, app, data…) serán las encargadas de llevar a cabo estas funciones e interactuarán con el código principal HTML del index.html que como hemos comentado antes será el encargado de montar la plantilla que rellenaremos con los datos que envíen dichos scripts.

Events.js será donde recogeremos las acciones que haga el usuario, es decir, aquí definimos cada botón de la página y le indicaremos que cuando el usuario pulse dicho botón se llame a una u otra función distinta dependiendo de lo que queremos que ocurra. Aquí está un ejemplo de código recogido de events.js para el botón Mapas de la página de inicio:

```
lng.dom('section#mapas  a[data-icon=search]').tap(function(event){
App.Map.buscacerca();
 });
```

Es muy simple, cuando pulse el botón indicado, que ejecute la función 'buscacerca' situada en Map.js. Este es el esquema básico de events.js

Data.js será el encargado de trabajar con la base de datos sql, trabajar con dichos datos, recogerlos o modificarlos y mandará a otras funciones que los recogerán para sacarlos en pantalla u otras necesidades.

Map.js hará todas las funciones relacionadas con GoogleMaps, como buscar la localización del usuario vía GPS, sacar un mapa en la pantalla o indicar una dirección.

View.js finalmente nos introducirá estos datos trabajados en la plantilla creada con HTML en el 'index.html' indicando el lugar exacto de la sección a la que nos hemos dirigido.

De esta manera ahorramos tener que dedicar una página específica para cada centro elegido, lo cual ocuparía demasiado espacio inútil ya que de esta manera solo tenemos que crear una plantilla vacía y rellenarla con Javascript en función de las elecciones del usuario.

6.5 Mapas

Esta es una de las herramientas más útiles que posee nuestra aplicación web. En los plugins de Lungo Sugar se incluye uno que hace las funciones de mapas y localización llamado GMap, pero nos daba bastantes problemas de configuración y su funcionalidad no era muy buena, de tal manera que decidimos crear un script nuevo llamado 'map.js' utilizando la API creada por GoogleMaps para desarrolladores de páginas y aplicaciones web.

Esta es una herramienta desarrollada por Google que permite a los creadores de páginas web incluir las funciones de Google Maps y Google Street View en sus páginas web de forma sencilla y sin tener que crear todo el código desde cero, ya que esta API ya tiene las funciones específicas creadas y es el propio Google el que trabaja con los datos, además este código es reconocido en cualquier página HTML ya que basta con definirlo como perteneciente a la API de Google.

El problema que nos encontramos es que la base de datos que estamos utilizando de OpenData no incluye las posiciones geográficas de los centros, por lo tanto lo que hacemos es modificar esta base de datos e incluir manualmente las posiciones de los 10 primeros centros de la lista, con el fin de ver el funcionamiento de los mapas y dar un ejemplo de cómo sería la aplicación web, ya que nosotros estamos creando una webapp que sirva de plantilla y ejemplo para futuros desarrolladores no vamos a perder el tiempo introduciendo manualmente las latitudes y longitudes de los 368 centros incluidos en la base de datos. En caso de que quisiéramos comercializar nuestra webapp sí que deberíamos hacerlo o ponernos en contacto con OpenData Navarra exponiendo nuestro proyecto y proponiéndoles que modifiquen la base de datos a nuestras necesidades.

En nuestra aplicación web utilizaremos esta herramienta de Google Maps para dos funcionalidades distintas:

-Posición de cada centro en un mapa: Accederemos a esta opción desde la sección individual que nos muestra las características del centro elegido como dirección, número de teléfono o localidad. Junto al apartado de dirección hemos añadido un botón llamado Mapa que nos dirigirá a un mapa en el que se muestre el punto donde se encuentra el centro elegido, para esto se extraerá la latitud y longitud únicamente de dicho centro y se mostrara dicha posición mediante la función de GoogleMaps.

Figura 18 – Botón para acceder al mapa del centro

Una vez pulsamos el botón que se muestra, los javascripts de Lungo se encargan de recoger la latitud y longitud del centro que se ha seleccionado y en la sección mapas mostrarnos el mapa con la posición de dicho centro:

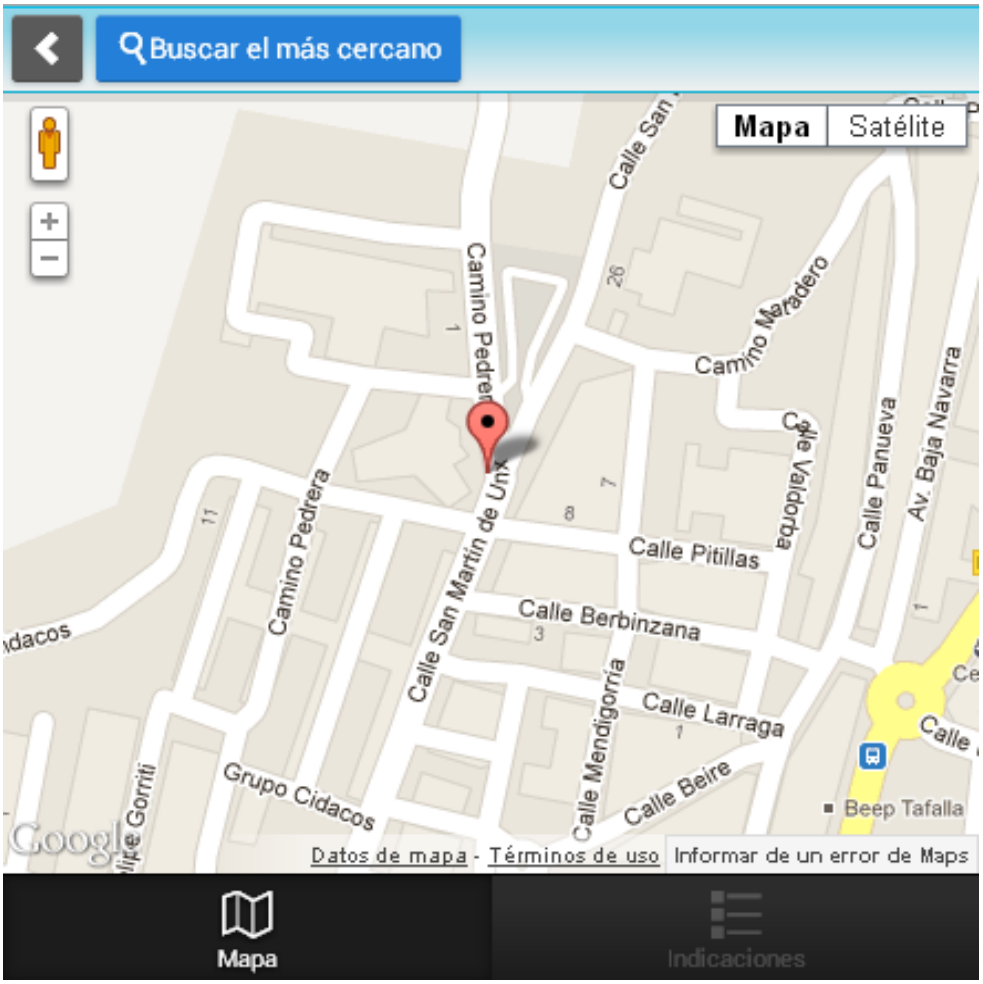


Figura 19 – Localización del centro

Esta es la primera utilidad para la que hemos utilizado la API de Google, no tiene mucha complicación aparte de conseguir seleccionar de la base de datos la latitud y longitud correspondientes al centro.

-Búsqueda del centro más cercano: Añadimos esta función a nuestra webapp porque en caso de querer comercializarla es de gran utilidad para los usuarios y aumentaría la funcionalidad.

Para acceder a esta herramienta de nuestra aplicación web se hay que dirigirse a la sección de mapas, o bien desde la página principal o bien desde cualquier mapa de un centro concreto. Si pinchamos en el botón mapas de la página principal, el navegador procede a localizar la posición del usuario y la muestra en un mapa:

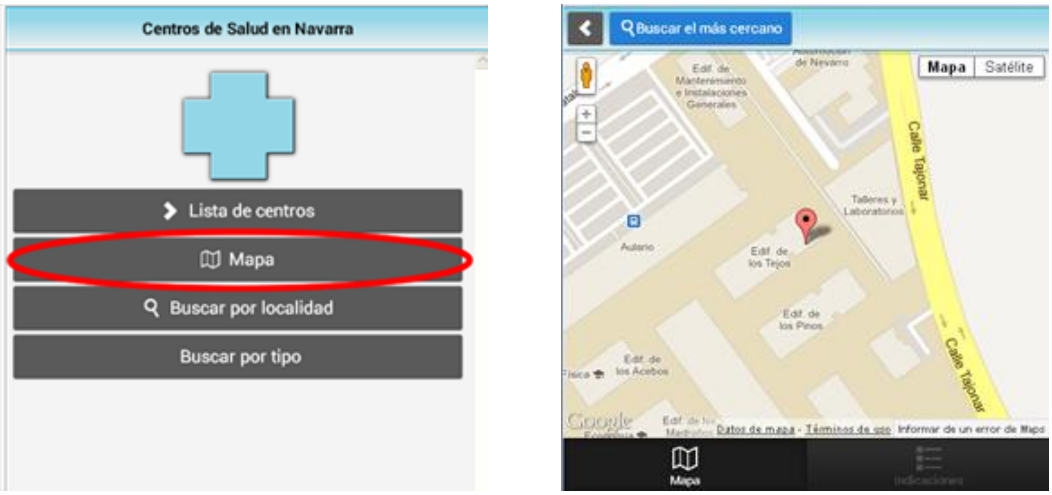


Figura 20 – Sección mapas

Para proceder a buscar el más cercano deberemos pinchar en el botón azul superior que pone ‘Buscar el más cercano’, entonces los scripts seleccionaran todos los centros de la base de datos y mediante una función de la API de GoogleMaps se halla la distancia mínima que hay a cada centro, se selecciona la distancia más pequeña de todas y se traza el recorrido entre la posición en la que se sitúa el usuario y el centro más cercano.

Nosotros hemos tenido que modificar esta función ya que no hemos introducido las 368 posiciones de los centros, únicamente de los 10 primeros para ver el correcto funcionamiento de nuestra aplicación web, como bien hemos explicado antes y a la vez para que sirva de ejemplo y ayuda a próximos desarrolladores que necesiten utilizar este esquema en su página.

Además GoogleMaps tiene una función que crea una lista con las direcciones que hay que seguir para llegar al destino final y la pondremos en el apartado 'Indicaciones' para facilitar a los usuarios llegar a dicho destino:

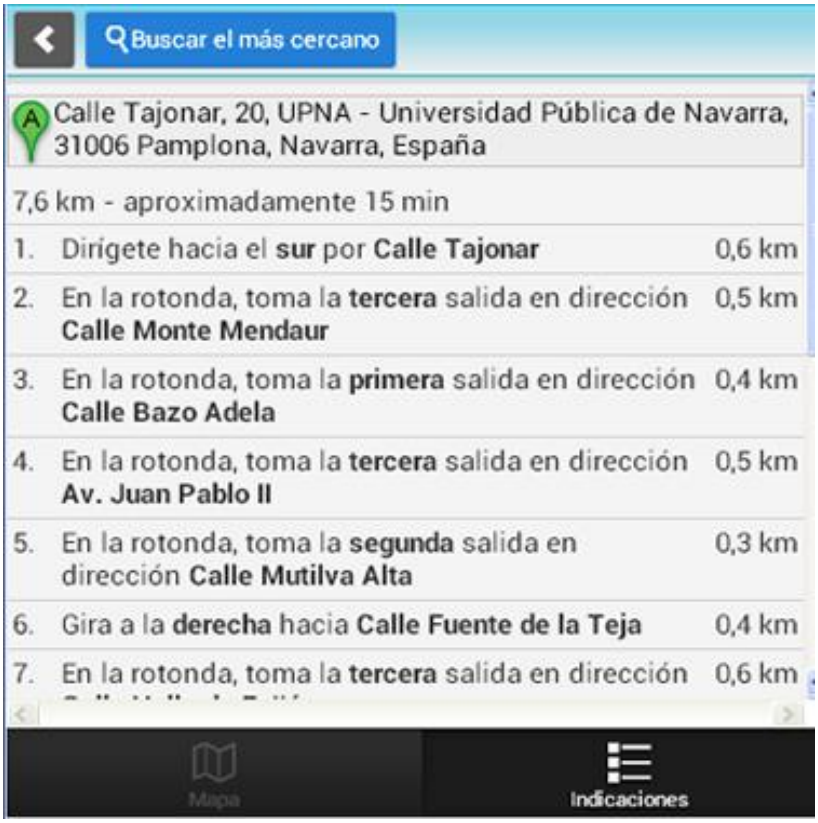


Figura 21 – Indicaciones de recorrido

Básicamente estas son las 2 maneras en las que hemos empleado las funciones de la API de GoogleMaps, concretamente hemos utilizado:

-DistanceMatrix(): Para calcular las distancias desde la posición del usuario hasta los diferentes centros y poder quedarnos posteriormente con la mínima.

-DirectionsService(): Para poder indicar las direcciones a seguir hasta llegar al destino, es necesario introducir las coordenadas origen y destino (esta última será la posición extraída de DistanceMatrix)

-getCurrentPosition(): Para que el navegador encuentre la posición en la que se encuentra el usuario vía GPS, para lo cual tiene que estar habilitada esta opción ya que hay algunos dispositivos que por seguridad la tienen deshabilitada.

Para entender y poder trabajar con estas funciones hemos necesitado estudiarnos detalladamente la documentación de la API de GoogleMaps ya que su lenguaje no tiene que ver con LungoJs ya que es individual de Google. Todo lo relacionado a estas herramientas que Google pone a disposición de los desarrolladores es libre y se puede encontrar en su página web:

https://developers.google.com/maps/documentation/javascript/

Y esto es básicamente todo en cuanto a las funcionalidades de mapas que hemos introducido en nuestra aplicación web.

6.6 Formulario

Hemos creado también un formulario de contacto para nuestra webapp situado en la sección de centros, la idea sería que si fuésemos a comercializar nuestra aplicación se incluyera en la base de datos la dirección de correo electrónico de cada centro para que los usuarios pudieran pedir cita por correo pero como no están incluidos lo que hemos hecho es que se dirijan todos los correos a un servidor web especial para correos que hemos habilitado.

El formulario que hemos hecho es muy básico ya que para nuestro caso no necesitábamos introducirle muchos parámetros ya que lo que queríamos es ver si se puede hacer con LungoJs y sirva de modelo. Por tanto le pedimos al usuario 3 parámetros, su nombre, dirección de correo y por último el mensaje que desea enviar por correo:

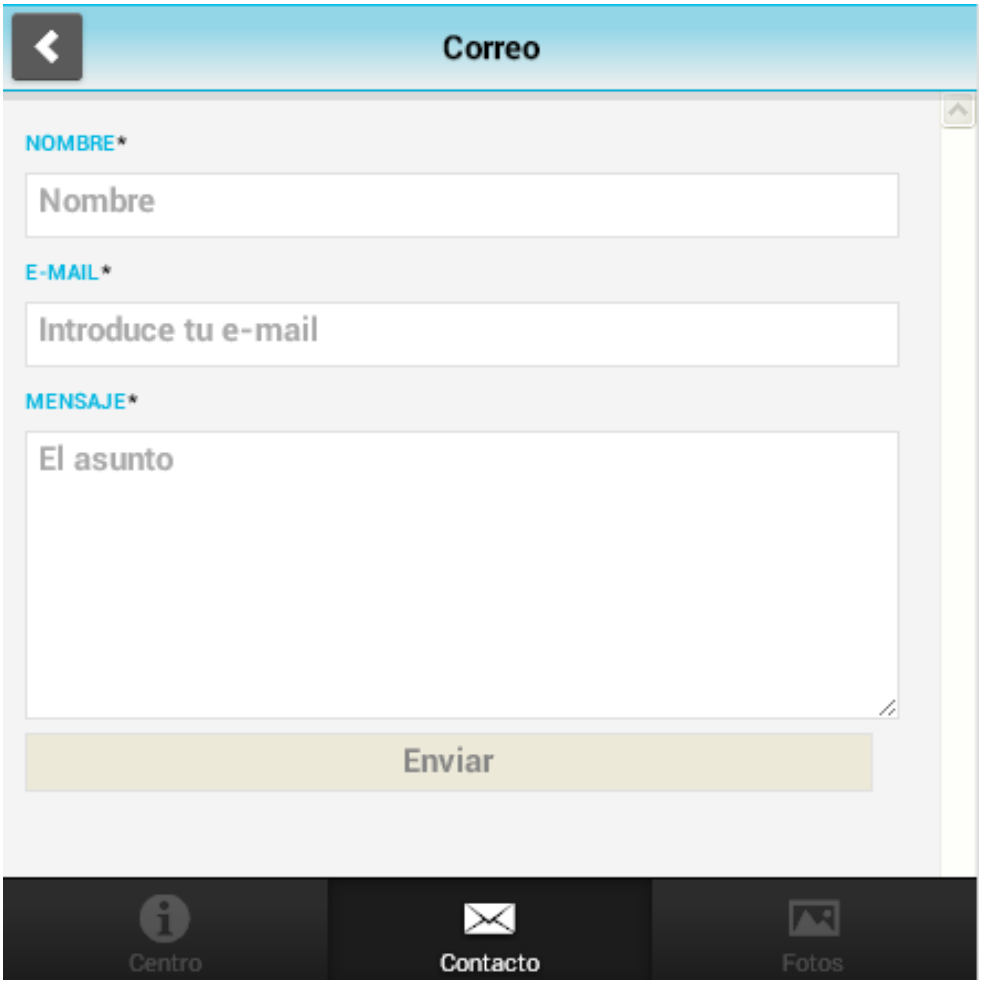


Figura 22 – Formulario de contacto

Para enviar un formulario de contacto es necesario usar PHP ya que el navegador no está capacitado para enviar correos, para esto es necesario habilitar un servidor especial de correo (servidor smtp) que hará las funciones de emisor del mensaje.

Lo que haremos entonces en el index.html será un formulario indicando que sea del método 'POST' de manera que el navegador ya sabe que será utilizado para correo.

Para que no se nos redirija ni se cambie la página al PHP y podamos seguir manteniendo todo como está vamos a realizar una petición Ajax con Javascript, por lo que creamos una función en el script Events.js de manera que cuando el usuario pulse el botón enviar, se recoja todo lo que está contenido en el formulario y se realice la petición Ajax al servidor utilizando el PHP que hemos creado con el nombre de mail.php.

```
$$.ajax({
    type: 'POST', // defaults to 'GET'
    url: 'http://rest',
    data: {user: 'soyjavi', pass: 'twitter'},
    dataType: 'json', //'json', 'xml', 'html', or 'text'
    async: true,
    success: function(response) { ... },
    error: function(xhr, type) { ... }
});
```

Figura 23 – Petición Ajax

Este es un ejemplo de cómo se hace una petición Ajax con LungoJs, nosotros le introduciremos como url el archivo 'mail.php'.

Nos hemos encontrado muchos problemas para conseguir el correcto funcionamiento de este formulario, Lungo trabaja básicamente con objetos, sin embargo por algún problema interno de configuración han implementado las peticiones Ajax de manera que no funcionen con objetos, por lo que tenemos que convertir los datos que vamos a enviar a un array, para lo cual hemos añadido una pequeña función que lo realiza.

Finalmente conseguimos que se mandaran los correos, aunque no funciona correctamente ya que en la consola de errores nos sigue marcando un error aunque funcione correctamente.

Para poder enviar el correo es posible hacerlo con una función de Php llamada mail() pero no es lo recomendable, ya que la mayor parte del tiempo no funciona o es sus mensajes son detectados como servidor por los correos.

Por eso hemos utilizado un complemento llamado PhpMailer que es una clase php para enviar emails que permite de una forma sencilla tareas complejas como por ejemplo enviar mensajes de correo con ficheros adjuntos o enviar mensajes de correo en formato HTML.

Se pueden enviar emails vía sendmail, PHP mail(), o con SMTP. Lo más recomendable es usando smtp porque con este complemento se pueden usar varios servidores SMTP. Esto permite repartir la carga entre varias computadoras, con lo que se podrán enviar un mayor número de mensajes en un tiempo menor.

Es posible enviar email con la función mail() de php, pero dicha función no permite algunas de las más populares características que proporcionan los clientes de correo usados actualmente. Entre estas características se encuentran el envío de email con ficheros adjuntos.

Por tanto utilizando la clase PhpMailer y un servidor smtp que hemos habilitado podremos enviar correos desde nuestra aplicación web.

En el php que utilizamos recogeremos los parámetros que necesitaremos para enviar el mensaje, es decir, el nombre, correo electrónico y mensaje del usuario. También deberemos indicar la dirección del servidor smtp que vamos a utilizar, si necesita usuario y contraseña los añadiremos y finalmente lo enviaremos.

Además se han añadido unas notificaciones para el caso de que el usuario envíe el correo sin haber rellenado todos los datos, de manera que le saldrá un mensaje indicando el campo que falta por rellenar.

Figura 24 – Notificación para completar campos

Finalmente si todos los campos están rellenados correctamente se envía el mensaje mostrando una confirmación y se puede seguir navegando por la aplicación sin que se redirija a ninguna otra página.

Figura 25

6.7 Galería de imágenes

Aunque para nuestra aplicación web no sea necesario incluir una galería de fotos, ya que no tiene que ver con el tema de centros de salud, hemos decidido introducir una para ver si es posible hacerlo con LungoJs y si funciona correctamente así como el abanico de posibilidades que ofrece este framework.

Buscando por la red encontrábamos numerosas galerías de fotos como complementos a aplicaciones web pero la mayoría son específicos para JQuery Mobile ya que es el framework más utilizado y desarrollado como hemos explicado en la introducción del proyecto, pero no servían para LungoJs ya que tiene un lenguaje de programación distinto.

Finalmente en el Sugar de Lungo (conjunto de scripts y plugins disponibles para este framework) encontramos un complemento llamado SimpleGallery que como indica su nombre, es una galería de imágenes muy sencilla.

Tuvimos que cambiar el código de este script y adaptarlo a nuestro gusto ya que funcionaba mostrando la galería en la página principal nada más abrir una página, pero nosotros queremos que la galería de imágenes este en una sección accesible desde cualquier centro u hospital elegido.

Figura 26 – Código modificado de la galería

Aquí podemos observar una línea de código que tuvimos que añadir para poder mostrar la galería en la sección deseada en lugar de que se mostraran las imágenes en pantalla completa nada más entrar en nuestra aplicación.

Como lo que pretendemos es ver si se puede añadir esta funcionalidad a nuestra webapp con LungoJs no vamos a complicarnos buscando fotografías para cada centro (aparte que seguramente no encontremos imágenes de casi ninguno) por tanto hemos escogido 5 fotografías que será con las que trabajaremos.

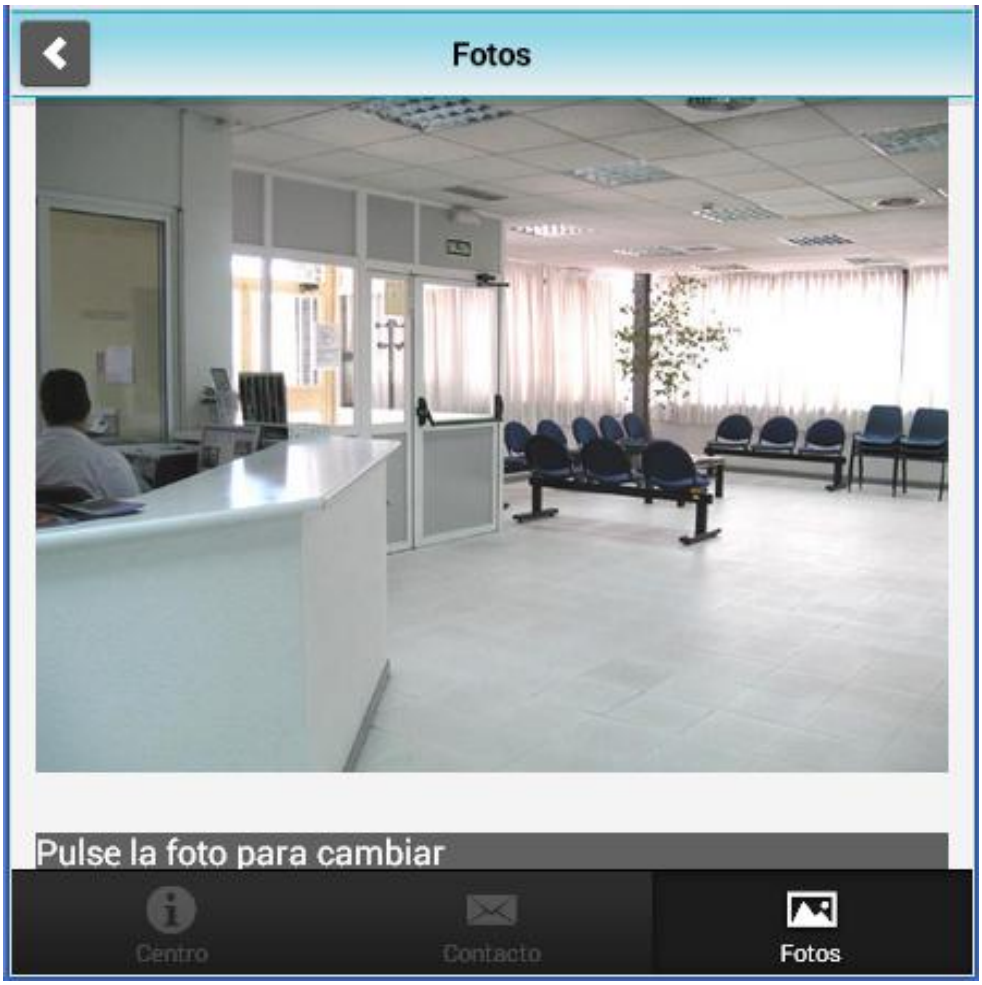


Figura 27 – Galería de imágenes

Basta con pinchar en cualquier parte de la imagen para que se cambie a la siguiente. Como se observa es una galería muy sencilla ya que no disponemos de una vista previa de las fotografías ni de un menú de navegación a través de ellas.

7. CONCLUSIONES

La primera conclusión a la que hemos llegado tras realizar este proyecto es que Open Data Navarra no ha respondido a las expectativas que se habían planteado sobre esta herramienta, ya que sus bases de datos no están actualizadas o están incompletas. Si se llegara a desarrollar más a fondo, podría ser una herramienta muy útil, como ha quedado patente en otros países en los que se utiliza, tales como Reino Unido y Alemania.

Esto se observa en que algunas de sus bases de datos simplemente se han creado pero no se han ido actualizando con el paso del tiempo por lo que los datos que las componen no sirven ya que están obsoletos, un ejemplo de esto es una base de datos que hay disponible sobre el estado de las carreteras, sin embargo, la última vez que se actualizó fue en 2011 por lo que su utilidad es nula.

Sí que hay que admitir que la idea que hay detrás de OpenData es muy buena siempre y cuando se desarrolle correctamente y se exploten todas sus posibilidades.

En cuanto al framework empleado, LungoJs, hay que decir que el desarrollo de aplicaciones web no es tan sencillo como se plantea, básicamente porque es un framework que todavía está en fase de desarrollo y por tanto tiene que pulir algunos detalles. Se ha tenido algún problema con la programación ya que no respondía como debería debido a pequeños fallos internos, no obstante todos estos pequeños problemas se han podido solventar.

También cabe destacar que al ser un framework que todavía está en fase de desarrollo no tiene muchos puntos de apoyo en la red, si se tiene algún problema apenas se dispone de información de casos similares en los que fijarse para solventarlo, al contrario que JQuery Mobile, del cual hay infinidad de páginas web de ayuda, complementos y tutoriales.

No obstante pese a estos pequeños inconvenientes cabe destacar que LungoJs es un gran framework para desarrollar aplicaciones web, es mucho más rápido que otros ya que el peso de sus archivos es mínimo (un punto a favor antes JQueryMobile) y a su vez el lenguaje de programación que emplea es muy intuitivo y ofrece las mismas posibilidades que el resto o incluso más que algunos frameworks.

Sin duda alguna se ha convertido en uno de los mejores frameworks para webapps pudiendo competir con otros más desarrollados y como opinión personal destacar que es muy recomendable emplear LungoJs antes que cualquier otro como Sencha Touch o JQuery Mobile que aunque ofrezcan más ejemplos y complementos, en cuanto a funcionalidad son similares a LungoJs sólo que mucho más pesados que éste. No hay duda que en cuanto se termine de trabajar y desarrollar LungoJs será uno de los frameworks más competentes del mercado.

Finalmente hay que fijarse que pese a que el tema de nuestra aplicación web eran centros sanitarios de Navarra, el esquema que se ha seguido puede aplicarse a cualquier otro tema.
Por esto podemos aplicar este proyecto como plantilla a seguir para cualquier persona que desee realizar una aplicación web conjunto a OpenData. Podría incluso llegar a ofrecerse como un proyecto de prototipo para realizar aplicaciones web utilizando OpenData ya que si se escoge otro tema como restaurantes, museos o farmacias se puede emplear este proyecto al completo ya que el esquema es similar en la mayoría de los casos y en este proyecto se han explotado todos los aspectos de los que se puede sacar partido en una webapp.

8. POSIBLES MEJORAS

Tras sacar nuestras propias conclusiones respecto al proyecto hemos pensado algunas posibles mejoras que se podrían añadir a éste y a sus herramientas:

-Nosotros hemos utilizado todo el rato la misma dirección de correo como destino de los formularios de contacto, pero una posible idea sería que cada centro de salud adjuntara su correo electrónico a la base de datos, de tal manera que el usuario de la aplicación pudiera ponerse en contacto con el centro de salud elegido por ejemplo para pedir una cita.

-A su vez se deberían adjuntar las posiciones de todos los centros elegidos para que el usuario pudiera obtener la posición de cualquiera de ellos a pesar de que se ya adjunte la dirección porque es lo que marcaría la diferencia para ser una aplicación web funcional y práctica.

-Podría usarse un programa como 'PhoneGap' o 'Titanium Appcelerator' para convertir nuestra aplicación web a una aplicación nativa para dispositivos móviles y facilitar su accesibilidad a los usuarios.

-La galería de imágenes no sería necesaria en nuestro caso, pero si se deseara podrían recopilarse imágenes de los diferentes centros y hospitales para añadirlos a la que se ha creado como prueba.

-El framework LungoJs necesita todavía unas cuantas mejoras ya que hemos encontrado bastantes problemas a la hora de utilizarlo. Por ejemplo el funcionamiento de las peticiones y servicios Ajax no están muy bien definidos y dan bastantes problemas, así como algunos elementos scrollables o la creación de plantillas para volcar los datos extraídos de la base de datos Sql interna, la cual es un punto a favor. Estamos seguros que todos estos problemas irán siendo solventados por los creadores de LungoJs que no paran de añadir mejoras a este framework en desarrollo.

9. BIBLIOGRAFÍA

-GOOGLE

-WIKIPEDIA

-LUNGO.TAPQUO.COM - Página oficial de LungoJs

- COMMUNITY.LUNGOJS.COM – Foro de preguntas de LungoJs

-SCREENCASTS DE JAVIER VILLAR – Conjunto de 3 videotutoriales del desarrollador y creador de LungoJs explicando su funcionamiento

-GITHUB.COM – Proyecto de base de datos de código libre para su uso público

-ENYOJS.COM – Página oficial de EnyoJs

-SENCHA.COM – Página oficial de Sencha Touch

-FOROS DE DESARROLLADORES WEB – Diversos páginas web de foros de preguntas y tutoriales de ayuda

-MANUALES HTML5, PHP Y JAVASCRIPT – Varios autores