

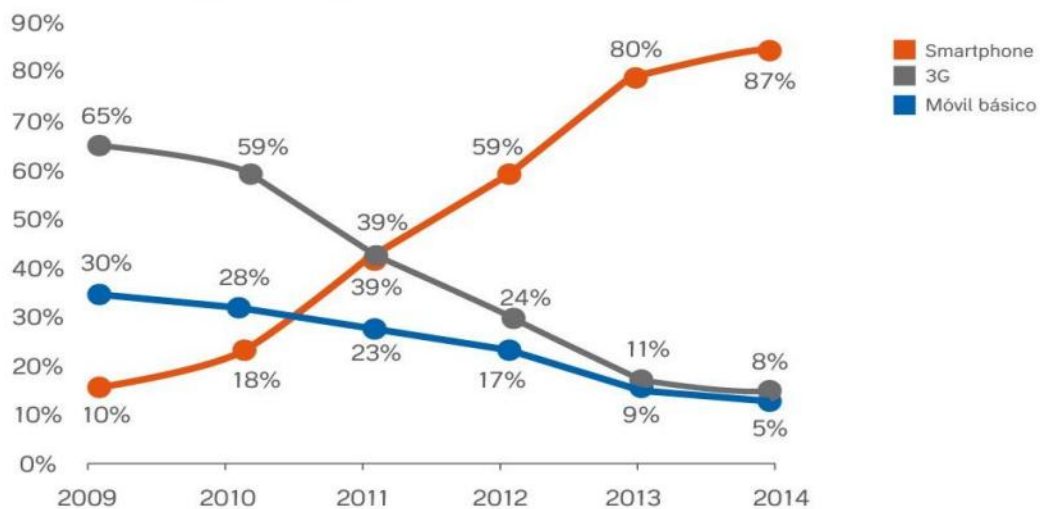
Introducción

El sector de la telefonía móvil actualmente es un sector en continua evolución, y a pesar de la recesión que comenzó en el año 2011, parece que el número de clientes de telefonía móvil es estable y con miras a crecer.

Según un informe publicado por Ditrendia en 2015 destacamos los siguientes puntos:

- Nadie usa más el smartphone que nosotros: España consolidó su liderazgo en el mercado de teléfonos inteligentes en Europa.
- Del total de líneas móviles activas en España, un 87% son smartphones.
- 7 de cada 10 españoles que posee un smartphone se conecta todos los días a través de este dispositivo, y más de la mitad durante más de treinta minutos.
- El 100% de los españoles que se conectan a Internet lo hacen también desde su smartphone y el 90% de los usuarios se conecta todos o casi todos los días.
- Las aplicaciones son la forma favorita para conectarse desde los dispositivos móviles: casi el 90% del tiempo de conexión se destina al uso de aplicaciones.
- El 80% de las empresas planea aumentar sus presupuestos de marketing digital en los próximos 18 meses, incluyendo el marketing móvil.
- Un 70% de los profesionales dicen que el móvil se convertirá en una herramienta fundamental para sus estrategias de marketing en los próximos cinco años.

Tamaño del mercado de Smartphones en comparación a los modelos básicos



Con esta premisa, y viendo el potencial del marketing en dispositivos móviles, es por ello que entendemos que orientar nuestro proyecto a una solución móvil relacionada con

el marketing sería un punto de partida ideal que pensamos, podría aplicarse perfectamente en un entorno real.

A día de hoy no es raro ver que una gran parte del comercio del panorama español no tiene una presencia demasiado significativa en el mundo del marketing online o marketing móvil, en especial aquellos comercios locales, que por tamaño, quizás no pueden permitirse la adquisición de soluciones móviles que ayudan a fidelizar a sus clientes. Es por ello que a menudo estos comercios pertenecen a asociaciones de comercio que realizan acciones de marketing conjuntas normalmente acciones de buzoneo y publicidad en medios de comunicación en papel, y es aquí donde entra en acción la solución que ofrecemos, mas adaptada a los tiempos que corren.

Razones por las cuales los consumidores quieren recibir (o no) mensajes de texto de las marcas



Gráfico elaborado por dirtrendia a partir de datos de Salesforce

A toda esta premisa hay que sumar el verdadero reto que nos proponemos, que es crear una aplicación multiplataforma que aproveche a su vez distintas funciones nativas del dispositivo (como la recepción de notificaciones, uso de gps, swipe) y poder hacerlo en un plazo de tiempo corto, cosa que es prácticamente imposible desarrollando la aplicación de forma nativa.

Análisis de soluciones

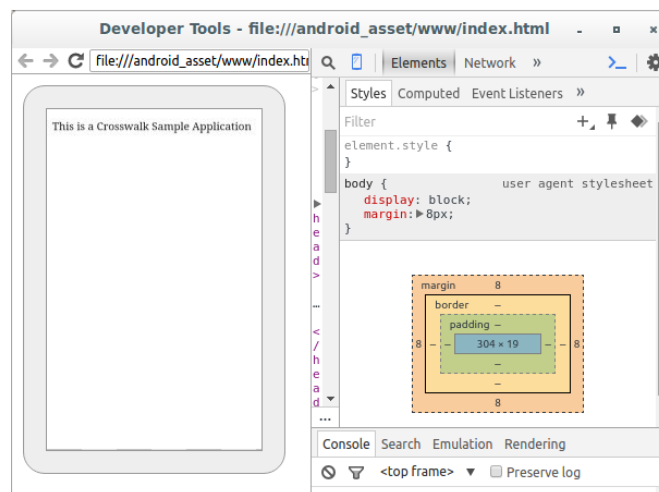
A priori, descartamos tecnologías nativas puesto que no nos permiten desarrollar una aplicación válida para todos los dispositivos en un plazo tan corto como el que requerimos y que a su vez tenga un coste atractivo para nuestro cliente objetivo, es por ello que nos vamos a centrar en soluciones híbridas que permitan que un solo código se ejecute en las tres principales plataformas móviles como son Android, IOS y Windows Phone, aunque éste camino también tiene sus hándicaps.

La ventaja que nos ofrecen las aplicaciones híbridas es a su vez su propia desventaja, y es que estas aplicaciones en su mayoría (a diferencia de Appcelerator Titanium) se ejecutan sobre el webview nativo del sistema del dispositivo, éste webview no puede acceder a la GPU del dispositivo como si lo hacen las aplicaciones nativas, por lo que puede resultar en un rendimiento deficiente si la aplicación tiene cierta carga de componente grafico o animaciones, así como un indeseable retraso en la respuesta al input del usuario.



Para solucionar este problema existe un plugin llamado Crosswalk que carga un webview basado en Chromium y que es capaz de hacer uso de las virtudes de WebGL como es acceder a la GPU del sistema y así favorecer el rendimiento de la aplicación.

A su vez, Crosswalk permite depuración remota utilizando las herramientas para desarrolladores de Google Chrome que van a facilitar mucho la ardua tarea que supone la depuración de lenguajes web.



Éstas son las opciones que hemos barajado:

- **Adobe PhoneGap**

PhoneGap es un framework para el desarrollo de aplicaciones móviles producido por Nitobi, y comprado posteriormente por Adobe Systems. Principalmente, PhoneGap permite a los programadores desarrollar aplicaciones para dispositivos móviles utilizando herramientas genéricas tales como JavaScript, HTML5 y CSS3. Las aplicaciones resultantes son híbridas, es decir que no son realmente aplicaciones nativas al dispositivo (ya que el renderizado se realiza mediante vistas web y no con interfaces gráficas específicas de cada sistema), pero no se tratan tampoco de aplicaciones web (teniendo en cuenta que son aplicaciones que son empaquetadas para poder ser desplegadas en el dispositivo incluso trabajando con el API del sistema nativo).



PhoneGap es una distribución de Apache Cordova, la diferencia principal entre Apache Cordova y Phonegap es que el segundo tiene acceso a servicios de compilación en la nube proporcionados por Adobe Creative Cloud.

Es un software de código abierto y por lo tanto puede ser utilizado libremente en cualquier aplicación sin necesidad de atribución o licencias de ningún tipo.

- **Ionic**

Ionic es una herramienta, gratuita y open source, para el desarrollo de aplicaciones híbridas basadas en HTML5, CSS y JS. Está construido con Sass y optimizado con AngularJS y, al igual que PhoneGap, Ionic es también una distribución de Apache Cordova.



El objetivo de Ionic es ejercer un modelo vista-controlador (MVC), un patrón basado en la ideología de separación de conceptos, que separa los datos, la lógica y las interfaces de usuario.

Entre las principales características de Ionic, es la inclusión de una interfaz de línea de comando, ya que podrás crear y compilar tus aplicaciones en cualquier plataforma. También tiene un alto rendimiento, a causa de una mínima manipulación del DOM (estructura de objetos que genera el navegador cuando se carga un documento), aceleraciones de transiciones y no trabaja con JQuery.

Uno de los rasgos más importantes y que caracterizan a Ionic es su diseño, ya que es muy sencillo de manejar, funcional, limpio (característica muy importante para los desarrolladores) y sobre todo optimizado para trabajar en casi todos los dispositivos móviles actuales. Una funcionalidad que ha sido incluida hace poco es la opción de poder desarrollar con el diseño de la interfaz Material Design, incluido en la versión de Android Lollipop 5.0 .

- **Appcelerator Titanium**

Appcelerator Inc es el nombre de una compañía privada especializada en tecnología móvil, con sede en Mountain View, California.



Sus principales productos son el IDE Titanium, un entorno de desarrollo de código libre para codificación de aplicaciones multiplataforma para dispositivos móviles, y la Plataforma Appcelerator, una suite de software y librerías en la que se basa el desarrollo, testeo, análisis, depuración y despliegue de tales aplicaciones. Dichos productos son frecuentemente agrupados, simplemente, bajo la denominación de Titanium o Titanium Framework.

En junio de 2009 se añadió soporte para el desarrollo de aplicaciones móviles para Android e iPhone. En abril de 2010 se añadió soporte para el desarrollo de aplicaciones para Ipad.

Las aplicaciones desarrolladas con esta tecnología se basan en el lenguaje Javascript, con uso de una API propia de la plataforma, la interpretación del código, desde 2011, se basa en el motor V8 Javascript engine de Google. Los componentes usados en la interfaz de usuario coinciden con los nativos de cada dispositivo.

- **Intel XDK**

Intel XDK es una herramienta para desarrollar apps cross-plataform utilizando HTML5. Con XDK, los desarrolladores pueden programar usando tecnologías estándar como HTML5 y desde una misma base de código generar apps para distintas plataformas. Con XDK es posible construir apps para las siguientes plataformas:



- Aplicaciones móviles: iOS, Android (nativo, Cordova, Crosswalk), Windows 8 Store, Windows Phone 8, Tizen, y Nook.
- Aplicaciones web: Web, Chrome App, Facebook App.

El XDK cuenta con un ambiente de desarrollo que permite emular apps en dispositivos virtuales para darse cuenta de cómo se verá su app en distintos dispositivos (iPhone, Microsoft Surface, Google Nexus, entre otros).

XDK también ofrece la capacidad de que los desarrolladores puedan almacenar su código en la nube de manera gratuita.

Gran parte de su atractivo radica en que, de la misma manera que podemos trabajar con el Intel App Framework por defecto, podemos trabajar con frameworks como Ionic o bootstrap, además de poder añadir plugins de PhoneGap o diferentes third parties para contar con funcionalidades como la recepción de notificaciones Push, complementándose todos estos frameworks y plugins perfectamente.

Otro gran aliciente es la posibilidad de añadir Crosswalk en la build de nuestra aplicación, aumentando el rendimiento general, eso sí, a costa de añadir casi 20Mb mas al peso total de nuestra aplicación.

- **Servicios web**

La aplicación necesita un servidor de aplicación que haga de intermediaria entre la base de datos y la vista de nuestra aplicación, además de aplicar cierta lógica de negocio, para ellos es necesario aplicar un patrón de diseño MVC (Modelo-Vista-Controlador), aunque como estamos tratando con una aplicación móvil, no se tratará de un patrón MVC estricto, dejando parte de la lógica del lado de la aplicación para así ahorrar consumo de datos móviles.

Para programar los distintos servicios que alojaremos en nuestro servidor de aplicación es necesario elegir entre las dos principales opciones que existen a día de hoy, **PHP** y **Node.js**.

- **Node.js**

Node.js es un framework, open source, que funciona del lado del servidor. Por lo tanto, permite que el servidor y las aplicaciones de escritorio se comuniquen por medio de Javascript.

Cuenta con su propio servidor y mantiene una única instancia del mismo, esto significa un uso mucho más eficiente de los recursos y una respuesta mucho más rápida a conexiones simultaneas... pero, no todo lo que brilla es oro, porque si un usuario ejecuta alguna tarea muy complicada puede provocar lentitud en todo el servidor y, por consecuencia, a todos los demás usuarios.

Node.js no es tan nuevo como muchos piensan pero su comunidad aún no es tan grande como la de PHP.

- **PHP**

PHP es un lenguaje de programación de uso general, open source, que funciona del lado de servidor. Originalmente diseñado para producir páginas web dinámicas, fue uno de los primeros lenguajes del lado de servidor que pueden ser embebidos dentro de HTML. El código PHP en un archivo tipo script que, al ser solicitado, es interpretado por un servidor web (generalmente Apache con un módulo procesador de PHP), el cual genera el contenido e imágenes dinámicamente. PHP puede ser

desplegado en la mayoría de los sistemas operativos, servidores web y plataformas.

PHP es un lenguaje muy maduro que cuenta con una gigantesca comunidad de programadores y desarrolladores experimentados.

Además, la comunidad de programadores PHP es muy generosa, la gente se ayuda mutuamente sin esperar nada a cambio y siempre hay tutoriales, artículos y soluciones a la disposición de cualquiera.

PHP corre sobre un servidor (Apache, Nginx u otros), donde cada conexión consume una pequeña parte de los recursos del servidor.

Esto es algo muy positivo porque significa que cada conexión es independiente del resto y, lo que un usuario hace, no afecta al resto (si un usuario realiza una tarea muy complicada, el sistema se pondrá lento solo para él).

Además, a lo mencionado anteriormente, hay que añadir que programar en Node.js requiere ser conscientes y estar constantemente pendientes de que cada función debe ser muy eficiente, de modo que si hay usuarios recurrentes trabajando, no noten lentitud... pero, en PHP, Apache resuelve esto automáticamente ya que cada llamada a PHP inicia una instancia del servidor, la cual resuelve la tarea sin afectar al resto y cuando termina la tarea, devuelve los resultados y libera los recursos.

Por lo mencionado anteriormente, creemos que a priori el uso de PHP es una opción más razonable.

- **Base de datos**

La base de datos es una parte esencial de un proyecto como éste en el que hay cierta interacción con el servidor a la hora de realizar por ejemplo, el registro de un usuario, login, carga de ofertas, etc...

Hay varias soluciones validas para este cometido, divididas en dos vertientes, bases de datos SQL y bases de datos NO SQL, con dos grandes representantes:

- **MySQL/MariaDB**

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base de datos open source más popular del mundo , y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web.

En 2009 se creó un ‘fork’ denominado MariaDB por algunos desarrolladores (incluido algunos desarrolladores originales de MySQL) descontentos con el modelo de desarrollo y el hecho de que una misma empresa controle a la vez los productos MySQL y Oracle Database.

- **MongoDB**

MongoDB es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON.

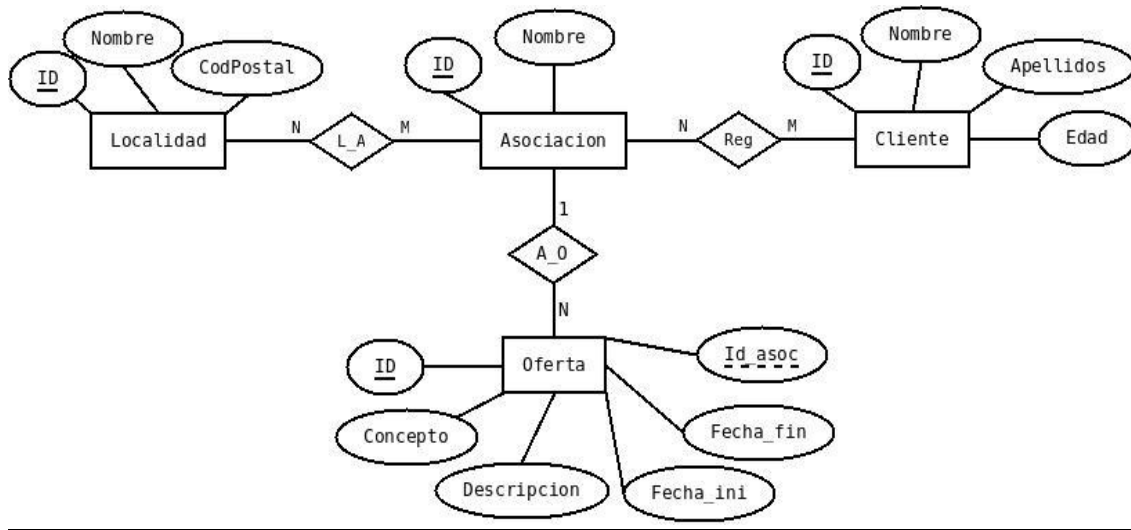
Una de las diferencias más importantes con respecto a las bases de datos relacionales, es que no es necesario seguir un esquema. Los documentos de una misma colección - concepto similar a una tabla de una base de datos relacional -, pueden tener esquemas diferentes.

Las bases de datos NoSQL son a priori interesantes, pero a la hora de crear una base de datos con relaciones es importante un diseño previo mediante un esquema relacional que ayuda mucho en su concepción, y esto con MongoDB es algo complicado.

Tampoco existen los JOINS. Para consultar datos relacionados en dos o más colecciones, tenemos que hacer más de una consulta.

Diseño y Arquitectura

- Esquema de base de datos



Una de las decisiones que hemos tenido que tomar a la hora de crear la base de datos es obviar una tabla '**Tiendas**' ya que supondría cargar las tiendas desde la base de datos cada vez que se ejecutara la aplicación, esto supone un gasto de datos móviles por encima de lo que creemos debe suponer una aplicación de estas características, todo ello en detrimento de una escalabilidad más sencilla a la hora de introducir o eliminar tiendas.

Todas las tablas cuentan con una columna de marca de tiempo de creación y id's autoincrementales para un funcionamiento óptimo de la bbdd.

Queremos destacar la tabla resultante de la relación N-M entre Asociación y Cliente, pues juega un papel crucial en el sistema de envío de notificaciones Push ya que en ella se almacena la relación entre el usuario registrado y una determinada asociación de comercio junto con un token identificativo (Google Cloud Messaging) que va a permitir que el usuario reciba notificaciones.

Para gestionar la base de datos hemos utilizado phpMyAdmin sobre MariaDB, todo ello instalado sobre un servidor alojado en hostinguer.es

- **Servicios**

- **Appservice.php**

Servicio al que se envía el formulario de registro mediante POST, éste a su vez devuelve una respuesta JSON con el caso correcto o el caso erróneo, según el caso la aplicación gestionará la respuesta y informará al usuario del resultado.

- **Applogin.php**

Servicio al que se envía el formulario de login mediante POST, éste a su vez devuelve una respuesta JSON con el caso correcto o el caso erróneo, según el caso la aplicación gestionará la respuesta y informará al usuario del resultado, la respuesta además devolverá datos de usuario que serán visibles en el submenú Cuenta de la aplicación.

- **Apptoken.php**

Servicio encargado de substituir el token del usuario por uno nuevo, este servicio se llama cada vez que arranca la aplicación o se hace login. Este script es necesario ya que Google Cloud Messaging recicla los tokens cada cierto tiempo, por lo que se deben renovar periódicamente.

- **AppPromos.php**

Este servicio va a devolver una respuesta JSON con el listado de promociones válidas según periodo de validez, ordenadas de mas actual a más antigua. Se llamará al pulsar el botón del submenú Promos.

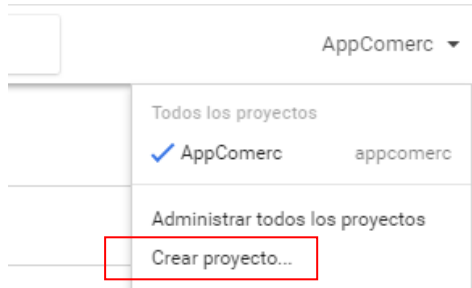
- **AutoNotis.php**

Script automático que se ejecuta el día 1 de cada mes, enviando una notificación a todos los usuarios.

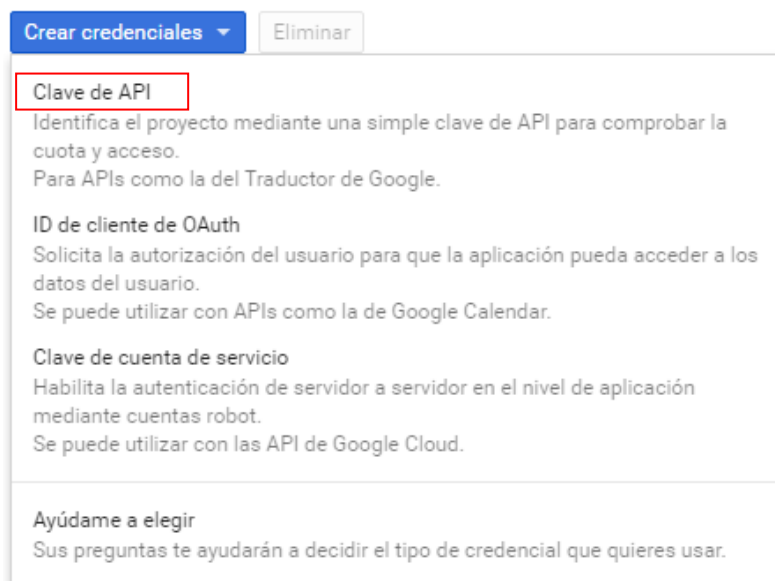
- **Google Cloud Messaging**

Para empezar, necesitamos tener una cuenta Google y acceder a <https://console.developers.google.com>.

En la parte superior derecha tenemos un menú donde deberemos crear un proyecto.



Una vez creado debemos definir unas credenciales



Al terminar obtendremos un número identificador de proyecto que utilizaremos desde la aplicación para obtener el token identificativo con el que se registrara luego el usuario.

Nombre del proyecto ?
AppComerc

ID del proyecto
appcomerc

Número del proyecto
44 733

Debemos crear una clave de servidor con la que podremos enviar las peticiones a GCM desde nuestro servicio. Si no especificamos una dirección IP, aceptará peticiones desde cualquier servidor (siempre y cuando tenga la clave).

Crear clave de API de servidor

Debes mantener en secreto esta clave en el servidor

Las solicitudes de API se generan mediante software ejecutado en una máquina que controlas. Los límites por usuario se aplican de forma obligatoria mediante la dirección que se encuentra en el parámetro `userIp` de cada solicitud, si se especifica. Si falta el parámetro `userIp`, se utiliza la dirección IP de tu máquina. [Más información](#)

Nombre

Clave de servidor 2

Aceptar las solicitudes de estas direcciones IP de servidor (Opcional)

Ejemplos: 192.168.0.1, 172.16.0.0/12, 2001:db8::1 o 2001:db8::/64

Dirección IP

Nota: Pueden pasar hasta 5 minutos antes de que se aplique la configuración

Al terminar el proceso tendremos un registrada nuestra clave.

Claves de API

<input type="checkbox"/>	Nombre	Fecha de creación ▾	Tipo	Clave
<input type="checkbox"/>	appcomerkey	18 may. 2016	Servidor	iYDZA7Pc

El registro del dispositivo seguirá el siguiente esquema:



1. La aplicación hace una petición al servicio Google Cloud Messaging con el numero identificador del proyecto.
2. GCM devuelve un token identificativo para ese dispositivo y la aplicación que ha hecho la petición.
3. Se registra usuario en la base de datos con los datos introducidos por el propio usuario, el identificador de asociación y el token identificativo obtenido, todo esto mediante el servicio appservice.php

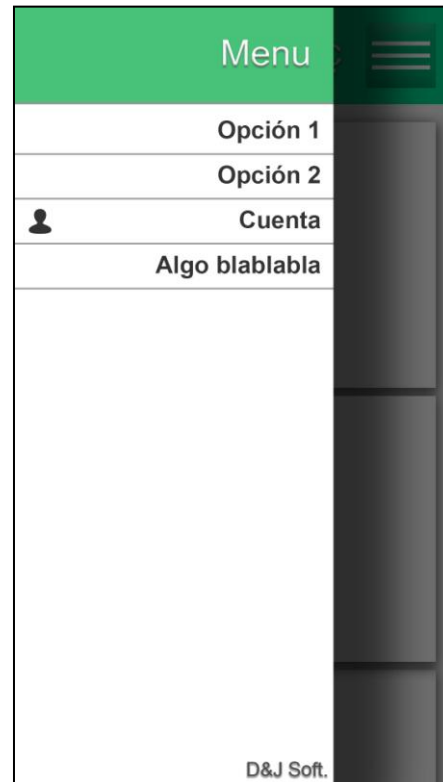
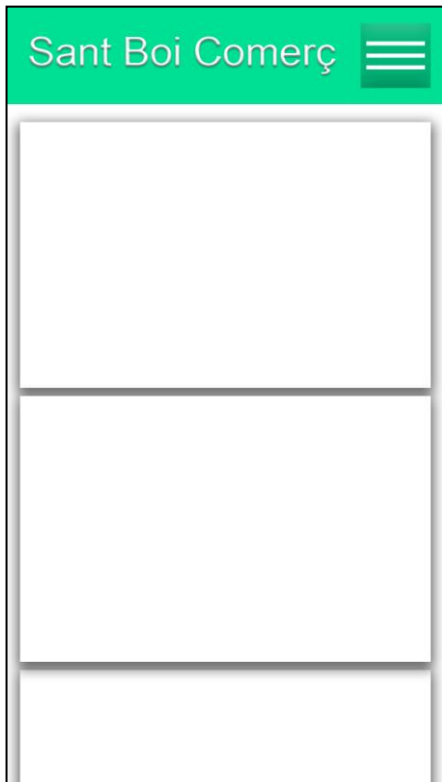


1. A través de un panel de notificaciones web o a través de un script automatizado, estableceremos el título y texto de la notificación, además indicaremos a que asociación queremos enviarla.
2. Una vez indicado todo esto, obtenemos los tokens de los usuarios de determinada asociación de comercio de la base de datos y con la clave de servidor obtenida anteriormente enviaremos a través de PHP CURL (en nuestro caso), una petición de envío al servicio GCM con todos los datos.
3. La notificación Push llega al dispositivo. La propia aplicación gestionará el evento de recepción de la notificación, y podremos ejecutar lo que consideremos necesario.

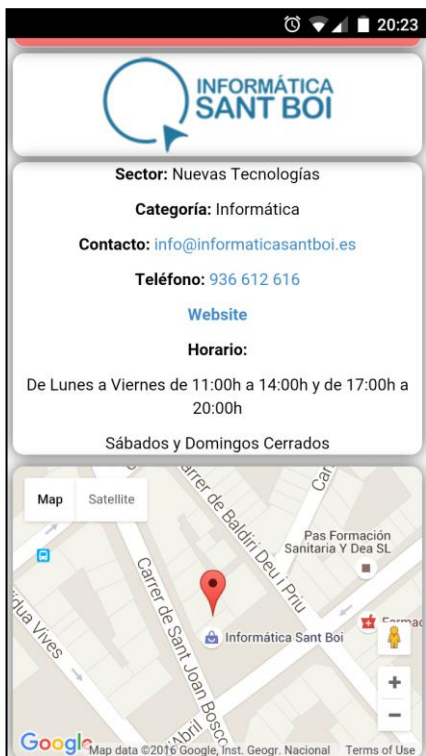
Importante: En aplicaciones Android hay que dar permisos a través del manifest, pero en aplicaciones híbridas, los plugins para notificaciones push ya hacen esto por nosotros.

- **Front-end**

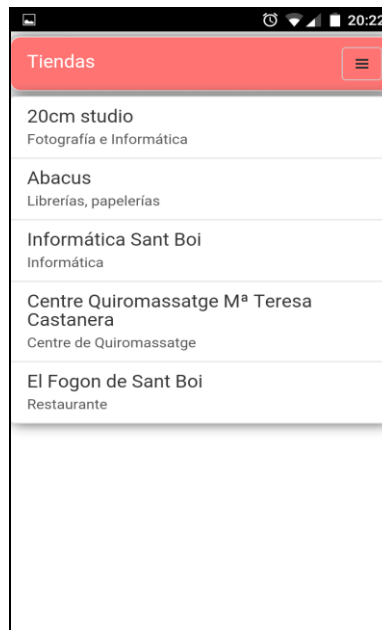
Diseño preliminar de la interfaz:



Dado que finalmente decidimos usar Twitter Bootstrap, el resultado final fue algo distinto, aunque nos ha servido para establecer una pauta a seguir.



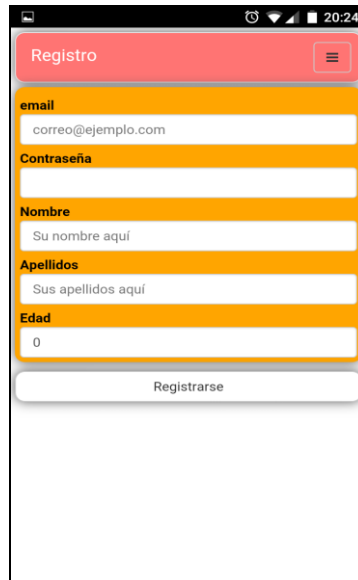
- **Características**
- Acceso a menú a través de botón hamburguer o a través de swipe (deslizando el dedo hacia la derecha).
- Submenú Tiendas: Carga un listview con las tiendas disponibles, al seleccionar una de ellas se accede a una vista de detalle del negocio con logo, información y widget de google maps con la situación de la tienda.



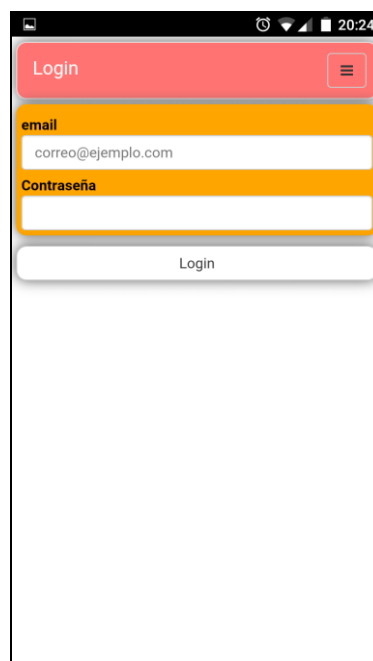
- Submenú Promos: Carga una vista de las promociones validas en el momento ordenadas de nuevas a antiguas.



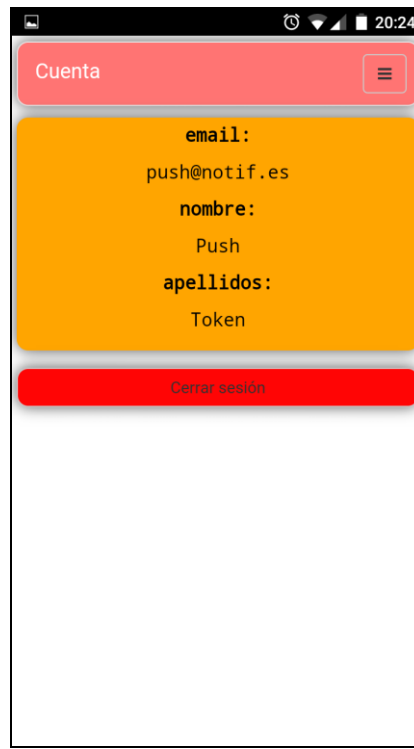
- Submenú Regístro: Muestra un formulario de registro, al dar al botón de registro se envían los datos del formulario a través de Ajax. El registro correcto cambia el menú, desaparecen los botones de registro y login para mostrar un botón Cuenta y se guardan los datos del usuario en el archivo Shared Preferences.



- Submenú Login: Formulario de login, al hacer un login correcto cambia el menú, desaparecen los botones de registro y login para mostrar un botón Cuenta, la respuesta del servicio de login devuelve los datos en JSON y se guardan los en el archivo Shared Preferences.



- Submenú Cuenta: Se cargan los datos del usuario guardados en Shared Preferences, además cuenta con un botón para cerrar la sesión del usuario actual.



Aspectos a mejorar

Al margen de nuevas e interesantes funcionalidades, el aprendizaje nos ha hecho darnos cuenta de que siempre hay cosas a mejorar, aspectos estéticos de la aplicación, implementación de servicios, utilización de nuevos frameworks, etc... estos son algunos de los puntos que creemos se podrían mejorar en siguientes versiones:

- Utilización de el framework Symfony para PHP con la api Orm Doctrine para trabajar con la base de datos.
- Implementación de notificaciones Push por geolocalización.
- Front-end mas atractivo para el usuario.
- Sistema de ofertas personalizadas para cada usuario, con creación de código QR personal como identificador de usuario, pudiendo usarlo en las tiendas para registrar compras y acumular descuentos.
- Buscador de tiendas, Categorías de tiendas.
- Fotogalerías dentro de cada comercio.
- Mailing automatizado.
- Comentarios de usuarios en cada comercio, sistema de ‘likes/megusta’.
- Compras desde la app.

Conclusiones

Trabajar sobre tecnologías web ha sido un reto para nosotros. A nivel de aprendizaje ha supuesto una carga importante, hemos tratado con lenguajes como Javascript o PHP que hemos tenido que aprender casi desde 0, además de los consiguientes frameworks y API's para el envío de datos de formulario, recepción de datos, eventos, etc...

También sacamos un punto importante y es que, la selección correcta de un entorno de programación adecuado como en este caso Intel XDK acelera el proceso y facilita mucho la tarea, sobre todo a la hora de hacer tests y depurar.

A pesar de las dificultades, los lenguajes y tecnologías que hemos usado cuentan con mucha información y documentación tanto oficial como de terceros, son tecnologías muy usadas en entornos reales y hay mucha gente que comparte dudas y soluciones en plataformas como Stack Overflow o GitHub.

En cuanto al proyecto en sí, el hecho de haber tenido que crear, tanto la aplicación como todo el entorno que lo rodea, servidor con sus servicios, base de datos, panel de control para notificaciones, notificaciones push etc..., todo ello nos ha dado una perspectiva más cercana a los procesos de negocio en el mundo del software

Bibliografía

- Intel XDK IDE - <https://software.intel.com/es-es/intel-xdk>
- Phonegap Push Notifications Plugin - <https://github.com/phonegap-build/PushPlugin>
- Google Cloud Messaging - <https://developers.google.com/cloud-messaging/?hl=es>
- Documentación PHP - <https://secure.php.net/manual/es/index.php>
- MariaDB - <https://mariadb.org/>
- W3Schools(Javascript, HTML, CSS, PHP) - <http://www.w3schools.com>
- Stack Overflow (Solución de errores) - <http://stackoverflow.com/>
- Hostinger.es - <http://www.hostinger.es/>
- Git - <https://git-scm.com/>

Agradecimientos

Queremos agradecer a todos aquellos que nos han echado una mano en la concepción de este proyecto. Eloy Albiach por echarnos una mano con tecnologías web, ajax y darnos un punto de vista sincero sobre los avances, Alex Maciá por ayudarnos en los primeros momentos en la concepción de una base de datos mínimamente coherente y en aquellos aspectos técnicos que han hecho que sea una base de datos más técnica y profesional, Juan Fco Gonzalez por proponer retos que han hecho rebanarnos los sesos y hacer que la aplicación sea más atractiva técnicamente, A Cristina Ara por hacer de betatester en la primera versión Alpha y animarnos a apurar los últimos días para aplicar las funcionalidades del submenú de promos, a Carles Romagosa por ayudarnos con el Front end en las fases previas y el diseño del logo de la app y por último a Verónica Díaz, Country Digital Manager en Nestlé Health Science que ayudó en el concepto que fue catalizador de todas las ideas aplicadas en este proyecto.