Transfer it easy

Aplicación para auto-relleno de formularios web mediante la lectura de código QR.

Alexander Nieves Parrales

Resumen— Cada vez es más habitual para las personas rellenar formularios en diferentes sitios web con sus datos personales y de contacto cuando se realizan compras en comercios electrónicos o cuando se registran en una web de su interés. Transfer it easy ofrece una alternativa para rellenar estos formularios con los datos personales y de contacto sin tener que escribirlos una y otra vez, agilizando así el proceso de registro y check-out al realizar compras en comercios electrónicos y a la vez evitar el abandono de la cesta de la compra provocado por procesos de check-out extensos y poco amigables para el usuario.

Palabras clave—Transferir datos, código QR, código bidi, aplicación Android, check-out, rellenar formulario, comercio electrónico.

Resum— Cada vegada és més habitual per a les persones omplir formularis en diferents llocs web amb les seves dades personals i de contacte quan es realitzen compres en comerços electrònics o quan es registren en una web del seu interès. Transfer it easy ofereix una alternativa per omplir els formularis amb les dades personals i de contacte sense haver d'escriure-les una i altra vegada, agilitzant així els processos de registre i de check-out al realitzar compres en comerços electrònics i, alhora, evitar l'abandonament de la cistella de la compra provocat per els processos de check-out extensos i poc amigables per a l'usuari.

Paraules clau— Transferir dades, codi QR, codi bidi, aplicación Android, check-out, omplir formulari, comerç electrònic.

Abstract— It is increasingly common for people to fill in forms on different websites providing their personal data or contact information when online shopping or signing up for a website they are interested in. Transfer it easy offers an alternative to filling in forms. It allows the speed up of the registration process and the check-out of online shopping, since the user does not have to write data over and over again. At the same time, it avoids the users' rejection of buying the purchases on their basket because of long and unfriendly check-out processes.

Index Terms— Transfer data, QR code, bidi code, Android application, check-out, fill the form, ecommerce, online sale.

1 Introducción

Cada vez que rellenamos un formulario estamos dedicando unos minutos de nuestro preciado tiempo a escribir nuestros datos personales, y es por eso que el objetivo de este proyecto es desarrollar una aplicación móvil que permita rellenar formularios web sin necesidad de escribir. A simple vista se denota el principal problema y una posible solución para el usuario final, pero este proyecto tiene un impacto que va más allá de ahorrar unos minutos al usuario rellenando un formulario web ya que existe una gran tasa de abandono de la cesta de la compra, que condiciona tanto las ventas de los comercios electrónicos (eCommerce en adelante) como las compras de los consumidores en línea (usuarios en adelante), dicho antecedente es causa de estudio en próximos apartados de este documento.

En la fase inicial la idea del proyecto pretendía abarcar el auto-relleno de formularios en el ámbito de registro y altas de usuarios en plataformas web, luego, junto con el tutor del proyecto creímos oportuno realizar una investigación sobre el proceso de check-out en eCommerce y el abandono de la cesta de la compra ya que este proceso requiere que el usuario introduzca datos personales, viendo en ello una posible oportunidad de uso para nuestro sistema.

El resultado de esta pequeña investigación dio como resultado una tasa de abandono de la cesta de un 69.23% en eCommerce. Y es que los usuarios que abandonaron la cesta generalmente lo hicieron en el proceso de check-out debido a la cantidad de información requerida, ésta tasa elevada de abandono nos motivó a ampliar la idea inicialmente propuesta y abarcar también el proceso de check-out en eCommerce.

Una vez definido el enfoque del proyecto realizamos un análisis de las metodologías ágiles y decimos que la mejor opción es utilizar el modelo de ciclo de vida del software conocido como desarrollo en espiral, debido a que el proyecto será desarrollado por una sola persona.

Este documento está conformado de los siguientes apartados en este orden:

- *E-mail de contacto: Alexander.Nieves@e-campus.uab.cat*
- *Mención realizada: Ingeniería del Software*
- *Tutor del proyecto: Rodolfo Guichón Aguilar*
- *Segundo semestre del curso 2016/2017*

- Introducción
- Estado del arte
- Objetivos
- Metodología
- Planificación
- Arquitectura
- Evolución y seguimiento
- Resultados
- Conclusiones
- Agradecimientos
- Bibliografía

2 Estado del arte

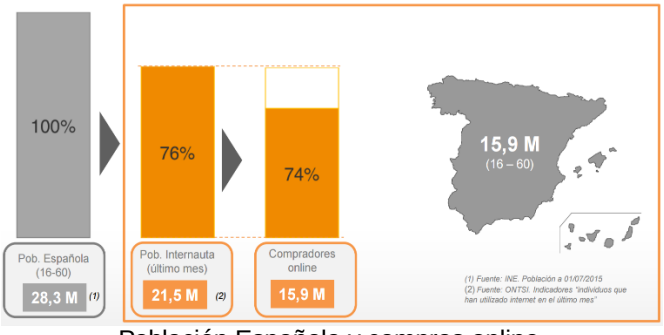
Según un estudio realizado en el último trimestre del año 2016 por Baymard Institute[1], se documenta en promedio un 69.23% de abandono de la cesta de la compra. Las principales razones entre otras son:

- Sitios webs requieren que el usuario se registre
- Procesos de check-out complicados y extensos.

Ambos motivos implican que el usuario introduzca un sin número de datos personales relacionados con su identidad, dirección de facturación, dirección de envío, datos de tarjeta bancaria (en el caso de check-out), etc.
Comprar en diferentes webs supondría al cliente repetir el proceso de introducir sus datos personales una y otra vez. Dichos antecedentes condicionan tanto las compras por parte de los usuarios como las ventas por parte del eCommerce.

En la actualidad el problema del abandono de la cesta provocado por procesos de check-out complicados y extensos se intenta solventar con la opción de comprar como invitado, lo que significa que los usuarios no deben registrarse pero deben introducir una mínima información relacionada con su identidad, la dirección de envío y datos de tarjeta bancaria. Para casos en los que los sitios webs requieren que el usuario se registre, se intenta requerir los mínimos datos durante el proceso de registro, desplazando la solicitud de otros datos personales al momento de realizar una compra.

Según el Estudio Anual de eCommerce del año 2016[2], en España, a fecha 01/07/2015 se registra una población de 23,3M de personas con una edad comprendida entre 16 y 60 años. De las cuales 21,5M de personas son internautas y 15,9M de ellas realizaron compras en eCommerce.



Población Española y compras online.

Existe una aplicación llamada Dashlane[3] que consiste en llenar formularios de manera automática utilizando una extensión para navegadores Chrome y Safari. Es necesario crear una cuenta y para rellenar un formulario web se hace uso de la extensión, por lo que si se desea rellenar formularios en otro dispositivo antes se ha de instalar la extensión e inicial sesión.

3 Objetivos

El principal objetivo de este proyecto es definir, diseñar y desarrollar un MVP (minimum viable product) del sistema Transfer it easy que permita rellenar formularios web mediante la lectura de un código QR con una aplicación para dispositivos móviles.
Para ello es necesario implementar 3 componentes principales:

1. Una aplicación para dispositivos móviles Android:
   - La aplicación ha de permitir almacenar datos personales de los usuarios de manera persistente.
   - La aplicación ha de permitir leer códigos QR haciendo uso de la cámara posterior del dispositivo móvil.
   - La aplicación ha de disponer de un método para enviar los datos personales del usuario después de leer un código QR.

2. Un servidor:
   - El sistema ha de disponer de un método para generar imágenes de códigos QR únicos.
   - El sistema ha de disponer de un método para recibir los datos de usuario desde la aplicación.
   - El sistema ha de disponer de un método para enviar los datos de usuario recibidos desde la aplicación a la página del formulario web del eCommerce.

3. Un widget:
   - El sistema ha de ejecutarse en el navegador web del usuario.
   - El sistema ha de solicitar una imagen de código QR al servidor.
   - El sistema ha de hacer visible la imagen de código QR en la pantalla del formulario web del eCommerce.
   - El sistema ha de recuperar los datos de usuario desde el servidor.
   - El sistema ha de escribir los datos de usuario en el campo correspondiente del formulario web.

3 Metodología

Después de realizar un análisis de las metodologías ágiles con las cuales se obtienen buenos resultados en Ingeniería de software, y teniendo en cuenta la naturaleza y necesidades del proyecto hemos decidido que la mejor opción es utilizar el modelo de ciclo de vida del software conocido como desarrollo en espiral Figura 1.

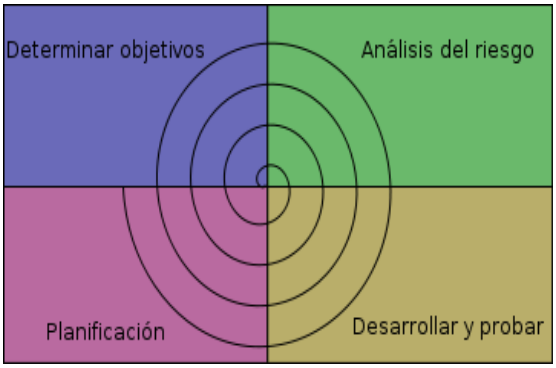


Figura 1. Desarrollo en espiral

El modelo de desarrollo en espiral nos permite llevar a cabo iteraciones en las que realizan el siguiente conjunto de actividades:

1. **Determinar objetivos:** Fijar requisitos, especificaciones y restricciones del elemento a implementar.
2. **Análisis del riesgo:** Estudiar las causas de los posibles eventos no deseados y el impacto que puedan generar gestión de riesgos), opcionalmente se pueden generar prototipos.
3. **Desarrollo y test:** Realizar tareas de Diseño, Código, Integración, Pruebas e Implementación.
4. **Planificar:** Revisar los resultados obtenidos y de acuerdo a esto determinar las tereas, tiempo y recursos necesarios para la siguiente iteración.

Por lo anterior expuesto y al ser un ciclo de vida orientado a gestión de riesgo concluimos en que es adecuado y se adapta perfectamente a las característica de este proyecto, y haciendo una correcta identificación de riesgos se puede alcanzar el objetivo con éxito.
Se mantuvo constante comunicación con el tutor (en ocasiones ejerce rol de Stakeholder) para de esta manera contemplar e integrar posibles cambios y mejoras tanto en el diseño como en la implementación del sistema. Esto es perfectamente compatible con el modelo de desarrollo en espiral.

4 Planificación

Para realizar una planificación perfectamente estructurada y gestionar el avance del proyecto hemos empleado la herramienta Microsoft Project, hemos decido desarrollar uno a uno los tres componentes del sistema en el siguiente orden:

1. Widget.
2. Servidor.
3. Aplicación para dispositivos móviles Android.

Para cada componente del sistema se realizan todas las actividades que implica seguir el modelo de desarrollo en espiral (determinar objetivos, análisis del riesgo, desarrollo y test, planificar) y subdividiendo cada componente en tareas más pequeñas dentro de cada una de las actividades correspondientes, esto permite realizar una mejor estimación del tiempo requerido y de la identificación de posibles riesgos para posteriormente establecer soluciones.
De cara al seguimiento hemos creído conveniente adoptar el concepto de Sprints de la metodología ágil SCRUM, para mantener una política de entregas frecuentes de software, en nuestro caso la duración de cada Sprint varía en función del componente que se está desarrollando y normalmente coincide con la actividad planificación donde nos reunimos con el tutor para revisar los resultados obtenidos.

A continuación se muestra la planificación inicial de los hitos con su fecha correspondiente:

Nombre	Fecha
Entrega componente Servidor Grizzly / Java	19/04/17
Entrega componente Widget	07/05/17
Entrega componente Aplicación Android	19/05/17
Entrega de proyecto y documentación final	27/06/17

Podemos observar que el primer componente en finalizar es el Servidor Grizzly pese a que antes hemos dicho que el primer componente a desarrollar es el Widget, esto se debe a que hay dependencias entre tareas de ambos componentes y por tal motivo no es posible finalizar el desarrollo del widget antes que el desarrollo del servidor, lo cual no implica que el orden de desarrollo no sea el antes expuesto.

5 Arquitectura

En la fase inicial, después de definir un diseño base del sistema hemos analizado posibles tecnologías a utilizar y elegido las que mejor se adaptan a las necesidades actuales y futuras del proyecto.

A continuación se muestra cómo interactúan entre sí todos los componentes del sistema Fig.2.

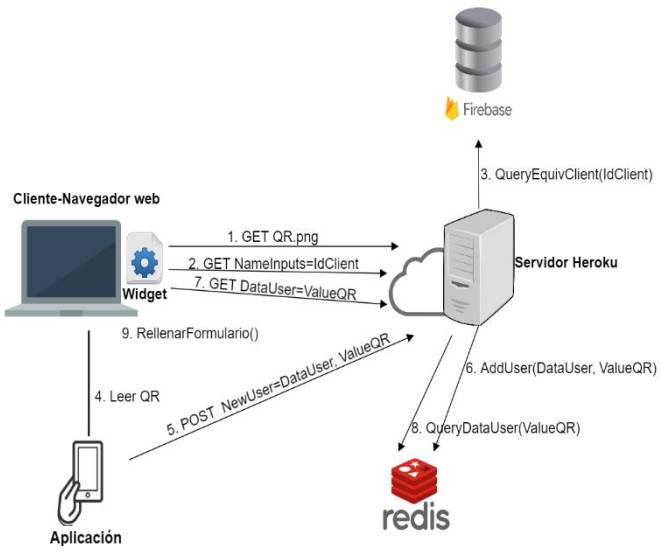


Fig.2 Arquitectura del sistema

Nuestro servidor utiliza la tecnología **Heroku** que es una plataforma como un servicio (por sus siglas en inglés PaaS) de computación en la nube que soporta varios lenguajes de programación, lo cual nos permite centrarnos en nuestra

aplicación y no en la infraestructura del servidor. Aquí alojamos nuestra aplicación de servidor ya que es una plataforma muy potente que nos ofrece un sinnúmero de herramientas que cubren las necesidades del proyecto:

- Deployment instantáneo: Realiza el build de la aplicación desde Heroku Git (consola de cliente heroku), una cuenta GitHub o Dropbox utilizando los ficheros del proyecto.
  En nuestro caso realizamos el deployment haciendo uso de una cuenta de Dropbox, aunque lo ideal sería utilizar GitHub para realizar un control de versiones más adecuado pero la opción de un repositorio privado es de pago frente a Dropbox que es de uso gratuito.
- Funciona sobre protocolo HTTPS lo que nos permite el intercambio de datos con el servidor de manera segura ya que los datos viajan cifrados a través de la red.
- Escalabilidad: Heroku utiliza el modelo de contenedor para ejecutar las aplicaciones, estos contenedores son llamados Dynos. La aplicación puede escalar a cualquier número determinado de Dynos en base a su demanda de recursos. La capacidad de gestión de Dynos proporciona una manera fácil de escalar y gestionar el número, tamaño y tipo de Dynos que la aplicación necesita en un momento dado.
- Add-ons: Gran cantidad de Add-ons que permiten añadir funcionalidad extra a las aplicaciones de forma rápida y sencilla, por ejemplo: Añadir una base de datos con unos sencillos pasos.
  Para nuestro proyecto empleamos el Add-ons Heroku **Redis** que es un almacén de datos clave-valor en memoria ram y que está a cargo de Heroku. Esta tecnología ofrece alta disponibilidad por lo que si la instancia principal falla, se reemplaza automáticamente con otra réplica que se mantiene en standby.
  Redis se encarga del almacenamiento de los datos de usuario recibidos desde la aplicación móvil, los cuales se almacenan de manera temporal (1 hora) y a continuación son eliminados. Hemos elegido Redis y no una base de datos NoSQL tradicional debido a las prestaciones mencionadas anteriormente y porque además proporciona un rendimiento muy alto (latencias muy bajas comparadas con otras bases de datos SQL o NoSQL) consumiendo menos recursos del sistema lo cual importante de cara a un número elevado de usuarios del sistema.

Nuestro servidor de aplicación que se aloja en Heroku despliega Jetty, que es un contenedor de Servlets y Servidor HTTP escrito en Java ultraligero con licencia Open Source, su condición de aplicación ligera la hace perfecta para embeberla dentro de nuestro servidor Java y ofrecer Servicios Web. Su funcionamiento se basa en la definición y utilización de un conjunto de componentes conectados entre sí:

- Una colección de Conectores (Connectors) que son los encargados de aceptar conexiones HTTP.
- Un conjunto de Manejadores (Handlers) que manejan las solicitudes provenientes de las conexiones, produciendo respuestas.
- Pool de Threads desde los cuales el servidor tomará recursos para realizar el trabajo.

El sistema requiere de almacenamiento persistente para los datos de nuestros clientes (eCommerce), la idea inicial era aprovechar Redis ya que también ofrece la posibilidad de almacenamiento persistente pero su coste es elevado por lo que elegimos emplear **Firebase** que ofrece un plan gratuito (100 conexiones simultaneas, 1GB de almacenamiento y un límite de 100GB de transferidos) suficiente para cubrir las necesidades iniciales de este proyecto. Es una base de datos NoSQL que está respaldada por Google ya que utiliza su infraestructura y es fácilmente escalable y además cuenta con una gran cantidad de información y documentación muy completa con ejemplos e incluso tutoriales.
El almacenamiento persistente es necesario para tener un registro de clientes eCommerce que utilizan nuestro servicio en sus sitios web y además para almacenar información relacionada a los inputs de cada formulario que el widget necesita a la hora rellenarlo.
El componente **Widget** se ejecuta en el Cliente-Navegador web[1] y es el encargado de una vez obtenido los datos del usuario y del eCommerce rellenar el formulario web. Se trata de un script escrito en lenguaje JavaScript debido a que es ligero, soportado por todos los navegadores modernos, conocido por el desarrollador y permite cumplir los objetivos de la funcionalidad.

El componente **Aplicación** es una aplicación para dispositivos móviles de sistema operativo Android escrita en lenguaje Java. Aunque lo ideal es contar con la aplicación en versiones para dispositivos Android y iOS debido a la duración del proyecto hemos decido desarrollar para Android que es una tecnología con la que el desarrollador tiene una mínima experiencia y dejar la versión de iOS fuera del alcance de este proyecto. Este componente es el encargado de dar al usuario la posibilidad de leer códigos QR, de almacenar los datos personales introducidos por el usuario y tenerlos disponibles para ser enviados al servidor después de que un usuario lea un código QR proporcionado en un formulario web de uno de nuestros clientes eCommerce.

6 Evolución y seguimiento

De acuerdo a la metodología del proyecto al finalizar cada sprint obtenemos un componente funcional, que es analizado juntamente con el tutor.

6.1 Componente servidor Jetty / Java

El primer hito del proyecto se dio el 07/05/2017 y la evolución del mismo es el adecuado según la planificación. Se consiguieron los objetivos establecidos y en la fecha estipulada:

- Diseño del servidor Grizzly / Java.
- Desplegar servidor en Heroku.

[1] El Cliente-Navegador web hace referencia a un usuario que requiere rellenar un formulario de un sitio web de uno de nuestros clientes eCommerce

- Instalación y configuración de Redis.
- Comunicación (cliente) eCommerce/servidor.
- Generar códigos QR con identificador únicos.
- Consulta de datos de usuario.
- Respuesta a petición de datos de usuario.
- Alta de usuario en Redis.

6.2 Componente Widget

El segundo hito del proyecto se dio el 19/04/2017 y la evolución del mismo es el adecuado según la planificación. Se consiguieron los objetivos establecidos y en la fecha estipulada:

- Comunicación (cliente) eCommerce/servidor.
- Autenticar eCommerce.
- Petición de los datos de usuario.
- Petición de código QR.
- Mostrar código QR.
- Rellenar formulario web.

6.3 Componente Aplicación Android

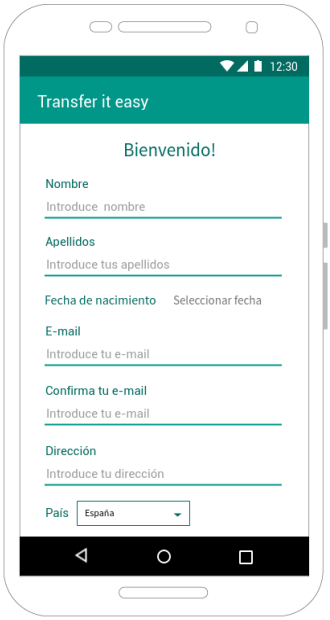
En este penúltimo hito del proyecto con fecha 19/05/17 la evolución del trabajo continúa siendo la adecuada como resultado del trabajo constante. Los objetivos conseguidos son:

- Fijar requisitos, especificaciones y restricciones.
- Mockup de app móvil.
- Diseño del app móvil Android.
- Formulario de datos de usuario.
- Almacenar de manera persistente los datos de usuario.
- Enviar datos de usuario al servidor.
- Leer código QR.

A continuación podemos observar el mockup de la aplicación que se diseñó durante este sprint.

Pantalla principal

Pantalla Capturar QR



Pantalla Mi Perfil

7 Resultados

Una vez finalizado el desarrollo de todos los componentes obtuvimos un producto funcional que cumple con los objetivos y requisitos establecidos inicialmente.

7.1 Servidor Jetty / Java

Un servidor alojado en la plataforma Heroku que procesa las peticiones del Widget y de la Aplicación Android y que interactúa según sea necesario con Firebase y Redis transfiriendo datos según formato JSON.

Un ejemplo de las funcionalidades del servidor sería en el caso de que el navegador realice la petición `https://transfer-it-easy.herokuapp.com/apiserver/equivalence?clientId=cli_myForm` para solicitar las equivalencias del eCommerce con identificador "cli_myForm". La respuesta será la siguiente:

```
{"name":"name",
 "suname":"snmes",
 "email":"nmail",
 "sexo":"sex",
 "confirm_email":"cnmail",
 "pass":"npass",
 "direction":"senddir",
 "postal_Code":"postal",
 "city":"local",
 "province":"prov",
 "country":"pais",
 "phone":"tel"}
```

Es decir, las equivalencias indican al widget que lo que almacenamos en el atributo "name" se debe escribir en el input con id "name" y lo que almacena el atributo "surname" se debe escribir en el input con identificador "snames" del formulario web, esto ocurre con cada atributo de la respuesta.

7.2 Widget

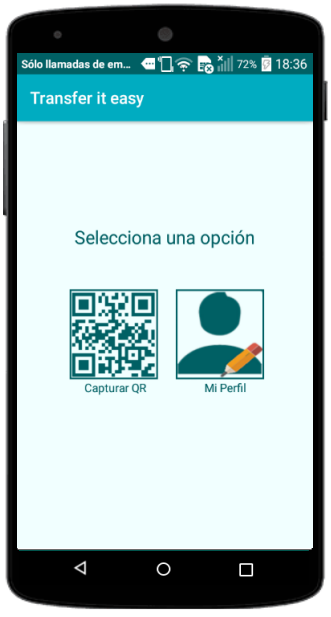
Consiste en un script escrito en lenguaje java que se ejecuta en el lado del cliente (navegador web) y que es accesible de manera externa para facilitar su mantenimiento y actualización. Este widget está accesible en un repositorio público de la plataforma GitHub. La siguiente imagen representa las líneas de código que el eCommerce debe incluir en la página del formulario web:

```
<div id=div_qr><img id="tritsyQR"></div>
<script src="https://rawgit.com/holanipa/tritsy/master/scriptTritsy.js"></script>
```

Como podemos observar es necesario un espacio donde mostrar la imagen del código QR, div id=div_qr.

7.3 Aplicación Android

Las imágenes que se muestran en este apartado son capturas de pantalla realizadas mientras se hacía uso de la aplicación real.

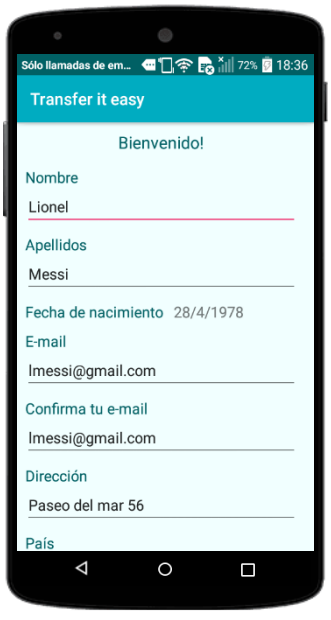


Pantalla principal

Una vez dentro de la aplicación se muestra la pantalla principal donde el usuario que la utiliza por primera vez tiene la posibilidad de introducir sus datos personales pulsando la opción Mi Perfil o leer un código QR haciendo uso de la opción Capturar QR.

La pantalla de la opción Mi perfil se muestra a continuación, desde aquí el usuario puede introducir sus datos personales, visualizarlos y modificarlos siempre que lo desee.
Se ha diseñado la interfaz de usuario estructurando los inputs de manera que sigan un cierto orden y se han utilizado componentes para facilitar la introducción de datos por parte del usuario, por ejemplo: El uso de DatePicker para la introducción de fechas y Spinners para seleccionar la provincia y ciudad de residencia. Estos componentes hacen que la interfaz sea más amigable.

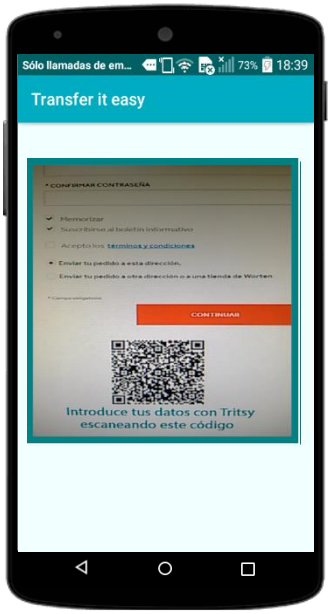
Selección de fecha de nacimiento



Pantalla Mi Perfil

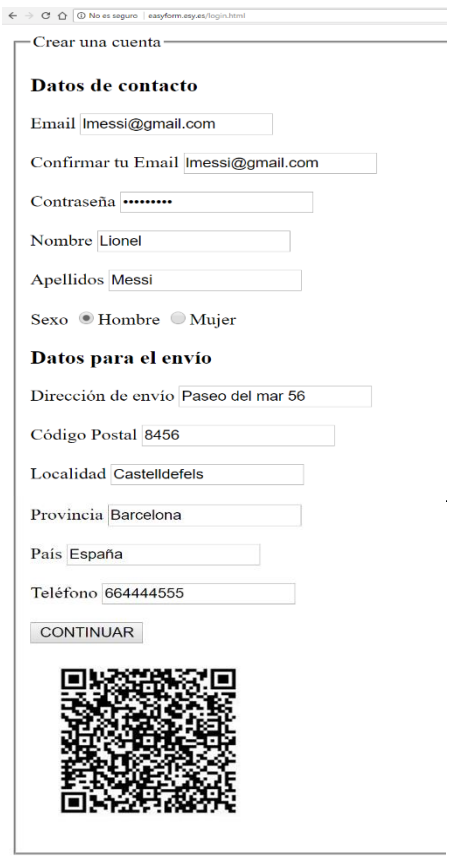
Pantalla Mi Perfil (cont.)

La otra funcionalidad de la aplicación es Capturar QR como su nombre lo indica permite capturar un código QR y obtener el valor que almacena, este valor actúa como un identificador que permite identificar los datos del usuario de manera única en nuestro sistema. Este identificador también lo utilizará el widget QR para recuperar los datos desde el servidor y posteriormente rellenar el formulario web. En la imagen a continuación se muestra la pantalla de esta funcionalidad.



Pantalla Capturar QR

Para realizar pruebas de la interacción de todos los componentes hacemos uso de un dominio gratuito donde alojamos un formulario web que simula un formulario de registro de usuario, dicho dominio está dado de alta en nuestra base de datos de clientes y puede interactuar con nuestro servidor. A continuación se muestra el formulario después de haber capturado el código QR con la aplicación y haber sido rellenado con los datos del usuario.



8 Conclusiones

Hemos conseguido alcanzar todos los objetivos propuestos y sin desviarnos de la planificación, ha sido una tarea muy laboriosa y de trabajo constante pero también ha sido una gran oportunidad de profundizar ciertos temas y adquirir cocimientos de tecnologías que antes eran totalmente desconocidas. La planificación realizada en la fase inicial del proyecto se ha seguido estrictamente, permitiendo cumplir con los objetivos y tiempos previstos. Debido al acierto en la metodología a emplear y al análisis exhaustivo del sistema que se realizó en la fase inicial los cambios realizados han sido mínimos por lo que el proyecto evolucionó de manera adecuada en todo momento.

Con nuestro sistema, a la hora de rellenar un formulario web se reduce el esfuerzo y tiempo significativamente (3 segundos aproximadamente, en el peor de los casos, desde la capturar del código QR). Aplicado a procesos de check-out puede ser un factor decisivo para que el usuario concrete una compra debido a la facilidad que Transfer it easy ofrece, en éste caso implica un incremento de conversiones para el eCommerce.

Transfer it easy participó en el concurso de STARTUP LAB 2017 (concurso de ideas en el ámbito de las TIC para estudiantes) en el que obtuvo el segundo premio, este es un motivo por el cual estamos contentos con el resultado final y además porque cumple con nuestras expectativas, el proyecto como TFG acaba aquí pero el proyecto como tal continuará en constante evolución.

9 Próximos pasos

- Formar parte de programa UAB Emprèn con el objetivo de mejorar el sistema para posteriormente tener disponible Transfer it easy en todo Internet.
- Desarrollar una versión de la aplicación para sistema operativo iOS.
- Permitir rellenar formularios web sólo haciendo uso del dispositivo móvil.
- Realizar ExploratoryTesting, que por limitaciones de tiempo no se ha realizado de manera exhaustiva.
- Restringir el acceso a la aplicación con un mecanismo de autenticación de usuario.

8 Agradecimientos

Para concluir este artículo quiero agradecer a Dios por permitirme vivir esta experiencia, a mí querida madre por ser un pilar fundamental y brindarme su apoyo incondicional. Agradezco también al Sr. Rodolfo Guichón Aguilar, tutor del proyecto, por los consejos y conocimientos transmitidos, y por el tiempo dedicado.

Doy las gracias también a todas las personas que indirectamente me ayudaron de una u otra manera y siempre estuvieron para darme ánimo y así no decaer.

9 BIBLIOGRAFIA

[1] Cart Abandonment Rate Statistics, , disponible en https://baymard.com/lists/cart-abandonment-rate (Consultado en febrero de 2017)

[2] Estudio Anual de eCommerce 2016, disponible en http://iabspain.es/wp-content/uploads/estudio-ecommerce-iab-2016_vpublica1.pdf

[3] Aplicación Dashlane, disponible en https://www.dashlane.com (Consultado en febrero de 2017)

[4] Heroku, disponible en https://www.heroku.com/ (Consultado en marzo de 2017)

[5] Grizzly/Java, disponible en: https://grizzly.java.net/ (Consultado en marzo de 2017)

[6] Desarrollo en espiral, disponible en https://es.wikipedia.org/wiki/Desarrollo_en_espiral (Consultado en marzo de 2017)

[7] Abandono del carrito de la compra, disponible en https://www.wearemarketing.com/blog/abandono-del-carrito-de-la-compra-por-que-ocurre (Consultado en marzo de 2017)

[8] Arquitectura REST, disponible en http://www.w3ii.com/es/restful/restful_quick_guide.html (Consultado en marzo de 2017)

[9] Ejecutar código REDIS online, disponible en https://redis.io/commands/hmset (Consultado en marzo de 2017)

[10] JavaScript de w3schools, disponible en https://www.w3schools.com/js/default.asp (Consultado en marzo de 2017)

[11] JSON Http Request, disponible en https://www.w3schools.com/js/js_json_http.asp (Consultado en marzo de 2017)

[12] CORS, disponible en https://www.html5rocks.com/en/tutorials/cors/ (Consultado en marzo de 2017)

[13] Acceso a objetos JSON en JavaScript, disponible en https://www.mkyong.com/javascript/how-to-access-jsonobject-in-javascript/ (Consultado en marzo de 2017)

[14] BASE64 encode bytes, disponible en http://iharder.sourceforge.net/current/java/base64/ (Consultado en marzo de 2017)

[15] RESTful, disponible en http://www.w3ii.com/es/restful/restful_quick_guide.html (Consultado en marzo de 2017)

[16] Ejecución asíncrona JavaScript, disponible en https://www.todojs.com/controlar-la-ejecucion-asincrona/ (Consultado en marzo de 2017)

[17] Carga asíncrona, disponible en https://rolandocaldas.com/html5/carga-asincrona-de-javascript (Consultado en marzo de 2017)

[18] Android - componentes de aplicaciones, disponible en http://www.w3ii.com/es/android/android_application_components.html (Consultado en abril de 2017)

[19] Android - Recursos de Organización y Acceso, disponible en http://www.w3ii.com/es/android/android_resources.html (Consultado en abril de 2017)

[20] Android – Documentación y ejemplos, disponible en http://www.w3ii.com/es/android/default.html (Consultado en abril de 2017)

[21] Android - REST y AsyncTask, disponible en http://www.arquitecturajava.com/android-rest-asynctask/ (Consultado en abril de 2017)

[22] Base de datos SQLite en Android, disponible en http://www.desarrollolibre.net/blog/tema/305/android/como-emplear-una-base-de-datos-externa-sqlite-en-android#.WSFJ7GjyjIU (Consultado en abril de 2017)

[23] Android - BarcodeDetector, disponible en https://developers.google.com/android/reference/com/google/android/gms/vision/barcode/BarcodeDetector (Consultado en abril de 2017)

[24] Android - CameraSource, disponible en https://developers.google.com/android/reference/com/google/android/gms/vision/CameraSource (Consultado en abril de 2017)