
# Recuperación de Información y Recomendaciones en la Web

30 de Noviembre de 2017

**Autores:**

Grupo 10

Martina De Luca - 4.904.514-4
Paula Villar - 4.769.924-2

**Docente:**

Libertad Tansini

## Contenido

## Introducción

En el presente documento se describe el conjunto de etapas abordadas para la construcción del sistema “15 primaveras”.

Se especifica el problema a abordar, el enfoque de solución propuesto, las funcionalidades del sistema implementado y sus limitaciones. Se presentan los resultados obtenidos y se incluye una sección en la cual se describen sugerencias de funcionalidades y/o mejoras a incluir en el sistema en futuras versiones del mismo.

Para la realización del obligatorio se identifica un problema el cual puede ser resuelto mediante la extracción de información disponible en la web, la cual luego de procesada y presentada al usuario, permite resolver el problema identificado. Se pretende además que el sistema a permita facilitar el proceso de toma de decisiones a los usuarios del mismo.

## Problema

Para la realización de este obligatorio se identifica como problema a resolver, la organización de fiestas de 15 años en Uruguay.

Existe en Uruguay una revista cuya edición es en papel, denominada Miss15 [1], que provee información de utilidad para realizar fiestas de 15 años. Tomando como referencia la bibliografía de Miss15, a la hora de organizar las fiestas es necesario contar, por ejemplo, con un salón de fiestas, servicio de catering, salón de belleza, fotografía, cotillón, entre otros.

La información de los diferentes servicios requeridos se encuentra disponible en la web, pero no en un sólo sitio, lo que dificulta o implica una búsqueda costosa en términos de tiempo para aquellas personas interesadas en la organización de fiestas de 15 años.

El sistema a diseñar pretende brindar información a aquellas personas interesadas en la organización de este tipo de eventos, de modo de dar soporte a la toma de decisiones sobre los servicios a contratar para la organización de la misma.

Actualmente, existen múltiples proveedores de este tipo de servicios, pero como fue mencionado, no es posible acceder a la información de dichos servicios de forma centralizada y sencilla, lo cual implica que aquellos usuarios interesados en organizar fiestas de 15 años deban dedicar mucho tiempo a obtener información sobre los servicios de su interés. Es objetivo de este obligatorio obtener la información de los servicios disponibles en la web, procesarla y presentarla a los usuarios de modo que resulte sencillo obtener la información que se requiere y ofrecerle al mismo herramientas que faciliten la toma de decisiones.

## Enfoque de la solución

En términos generales, para ofrecer una solución al problema planteado, se propone identificar fuentes de datos que provean la información relativa a los servicios mencionados, extraer dicha información, procesarla, almacenarla y construir una aplicación web que ofrezca la información obtenida al usuario.

### Investigación

Se realiza una etapa de investigación en la cual se identifican las fuentes de datos. Se investigan: Servicios MercadoLibre [2] , El Gallito Luis [3], Páginas amarillas [4], UruguayTotal [5], Tu fiesta [6] y la información provista por Google Maps al buscar un servicio.

Se seleccionan como fuentes de datos a extraer: el sitio de páginas amarillas y la información provista por Google Maps para obtener los datos necesarios a incluir en la solución. La selección de las fuentes fue realizada en función de la calidad de los datos disponibles; es decir la cantidad de información/atributos disponibles del servicio (nombre, teléfono, dirección, calificación) y la estructuración de los datos.

La siguiente imagen ilustra, en términos generales, el proceso de extracción, transformación y carga realizados.


Figura 1: Proceso de extracción, transformación y carga

### Extracción

Para realizar la extracción de datos se investigan herramientas de scraping disponibles en la web y se propone utilizar dexi.io [7]. Con esta herramienta, es posible seleccionar, de los sitios web elegidos para realizar la extracción, aquellos atributos de interés y la herramienta automáticamente realiza la extracción de los mismos brindando como resultado un conjunto de datos en formato csv.

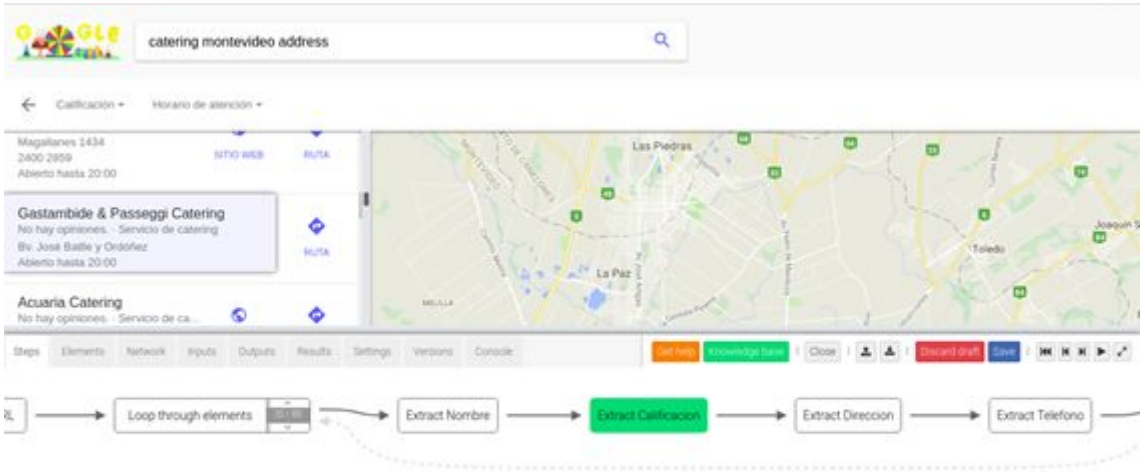


Figura 2: Ejemplo de extracción de datos con dexi.io en Google Maps

## Transformación y carga

Como se menciona en más detalle en la siguiente sección, se utiliza ElasticSearch como motor de búsqueda para implementar la solución, pero también como base de datos en sí misma, por lo que los datos extraídos son almacenados en la instancia de ElasticSearch utilizada.

ElasticSearch provee de una API a través de la cual es posible ingresar el conjunto de datos del sistema en formato JSON. Como se menciona en el punto anterior, dexi.io retorna el conjunto de datos extraido en formato .csv por lo cual es necesario transformar el formato; para ello se utiliza la herramienta FreeFormater [8] que permite transformar un archivo .csv en .JSON.

De todos modos, al extraer datos de dos fuentes diferentes, si bien la semántica de los mismos coincide no sucede lo mismo con su estructura. Por ello fue necesario definir una estructura común y realizar las transformaciones correspondientes.

Los problemas identificados son:

- Los datos extraídos del sitio web de páginas amarillas cuentan para cada servicio con los atributos: nombre, dirección, barrio, ciudad y teléfono. Sin embargo, no se encuentran disponibles bajo tales nombres de atributos.
- Los datos extraídos de Google Maps cuentan, para cada servicio con los atributos: nombre, dirección, teléfono, barrio y calificación. Sin embargo, no se encuentran disponibles bajo tales nombres de atributos.

Por lo antes mencionado, luego de la extracción, fue necesario asignar un nombre de atributo a los datos extraídos y este proceso de identificación y asignación fue realizado de forma manual.

Además, no todos los servicios cuentan con todos los datos o algunos de ellos contienen en el campo ciudad el nombre de la ciudad y el barrio; para estos casos fue necesario una transformación manual. También fue necesario modificar el formato de algunos atributos, como por ejemplo, la eliminación de espacios al comienzo o final de los valores.

Para facilitar el proceso de búsqueda y filtrado se decidió transformar el valor de calificación de los servicios al número entero más cercano, evitando así tener datos con valores decimales.

### Arquitectura

Para la implementación del sistema se propone desarrollar una aplicación web a través de un proyecto Maven en JavaScript. La arquitectura propuesta es cliente/servidor.

Como fue mencionado en la sección anterior se utiliza ElasticSearch como motor de búsqueda pero también como base de datos. Esto se debe a que es necesario almacenar los datos en la instancia de ElasticSearch para realizar la búsqueda y filtrado. Por lo tanto, no es necesario tener otra base de datos con esta información, como se había propuesto anteriormente.

Como servidor se utiliza Apache Tomcat [9], a través del cual se puede acceder a la aplicación implementada.

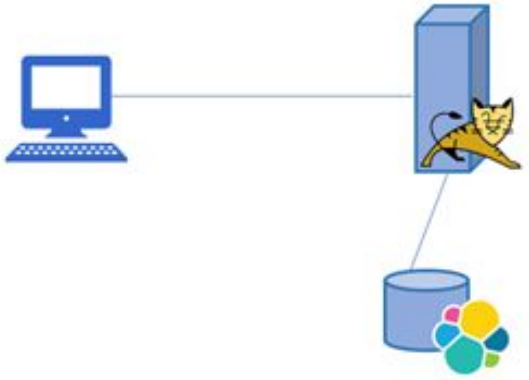

Figura 3: Arquitectura de la aplicación

## Implementación

Como se menciona anteriormente, se utiliza la herramienta dexi.io para la extracción de datos de las páginas seleccionadas. Esta herramienta permite la construcción de “robots” para cada página, a los cuales se les indica qué campos extraer. Esta herramienta permite también recorrer y extraer datos de las distintas páginas de resultados que pueda tener un sitio. Estos “robots” implementados se almacenan en el sitio de dexi.io, los cuales pueden ser reutilizados para una nueva extracción de datos, si es necesario.

Una vez extraídos y transformados los datos, estos fueron almacenados directamente en la instancia utilizada de ElasticSearch.

Para la implementación del sistema se crea un proyecto web Maven en Java. Este tipo de proyecto permite resolver las dependencias existentes con mayor facilidad.

Uno de los problemas a resolver en la implementación es el acceso a los datos almacenados en la base de datos de ElasticSearch. Este producto provee clientes para Java, JavaScript, .NET, PHP, Python y Ruby. Se utiliza en este caso el cliente para JavaScript, elasticsearch-js [10]. Por lo tanto, se desarrolla la aplicación en JavaScript para el acceso a los datos almacenados en ElasticSearch.

Para construir la página web se utiliza HTML y CSS junto con la librería Leaf [11]. Esta librería provee un conjunto predefinido de estilos y se utiliza con el fin de obtener un conjunto estándar de estilos para la página. De esta forma, se obtiene un diseño homogéneo para la aplicación.

## Funcionalidades y uso

Al ingresar al sitio web de quince primaveras se despliegan todas las opciones de servicios disponibles (servicios de catering, repostería, fotografía, peluquería, etc.) de modo que el usuario puede ver los datos asociados a cada servicio al ingresar.

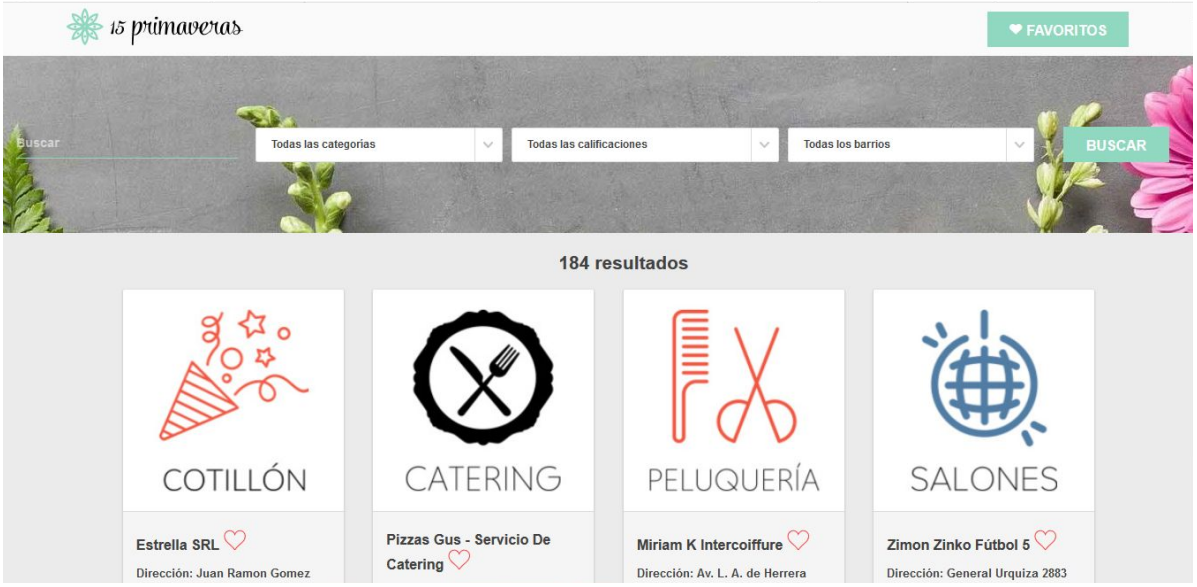


Figura 4: Página principal de la aplicación

En las siguientes secciones se detallan las funcionalidades implementadas.

## Búsqueda y filtros

En la parte superior de la pantalla se dispone un campo de texto para realizar la búsqueda y los filtros disponibles, junto con el botón de “Buscar” que permite realizar la búsqueda especificada. A continuación se detallan estas funcionalidades.

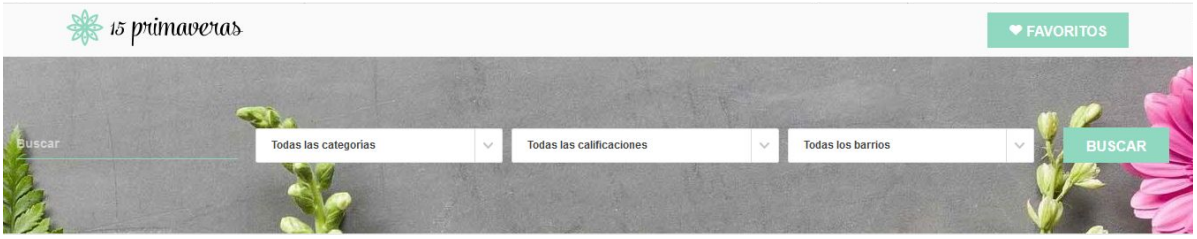


Figura 5: Buscador y filtros

- Buscador: Los usuarios pueden realizar búsquedas generales sobre los servicios, tanto por su nombre, dirección o ciudad. El sistema retornará aquellas opciones que contengan en alguno de estos atributos el texto ingresado en el buscador.

- Filtros: Además de las búsquedas los usuarios pueden filtrar los resultados presentados por:
    - Categoría
        - ➔ Catering, Cotillón, Fotógrafos, Peluquería, Repostería, Salones, Vestidos
    - Calificación
        - ➔ 1 a 5
    - Barrio
        - ➔ Atahualpa, Barrio sur, Belvedere, Buceo, Capurro, Carrasco, Centro, Cerrito, Ciudad Vieja, Colón, Cordón, Ituzaingó, Jacinto Vera, Larrañaga,Lezica, La Blanqueada, La Comercial, La Teja, Mercado Modelo, Manga, Malvín, Palermo, Parque Batlle, Parque Rodó, Paso de las Duranas, Paso de la Arena, Peñarol, Piedras Blancas, Pocitos, Prado, Punta Carretas, Punta Gorda, Reducto, Sayago, Tres Cruces, Tres Ombúes, Unión y Villa Muñoz.

  Al seleccionar la opción para filtrar, los resultados se construyen como la intersección de aquellos que cumplen con los filtros seleccionados; es decir si el usuario desea filtrar por la categoría “Peluquería” y barrio “Pocitos” los resultados a desplegarse serán todos los servicios de tipo Peluquería cuyo atributo barrio se corresponde con Pocitos.

- Búsqueda y filtrado: Es posible además, a los filtros aplicados, buscar cierto texto particular. A modo de ejemplo, un usuario puede interesarse por aquellos servicios de catering con calificación 4 y recibió recomendaciones de un servicio de este tipo que recuerda contenía la palabra “Rey” en su nombre. Este tipo de búsquedas también son posibles en el sistema implementado.

Los filtros disponibles permiten que los usuarios no solo puedan obtener información sobre servicios particulares sino también aquellos cuya ubicación en la ciudad les resulte conveniente filtrando por barrio de la ciudad de Montevideo.

## Calificaciones

Como fue mencionado, algunos servicios tienen una calificación asociada la cual se genera en base a recomendaciones de usuarios reales de los servicios mencionados. Tales recomendaciones son recolectadas por Google, quien al detectar que un usuario se encuentra en un sitio que corresponde con la dirección de uno de los servicios consulta cuál es su opinión al respecto.

Contar con esta información permite al usuario tomar decisiones en función de recomendaciones de usuarios anteriores.

## Favoritos

Por otra parte, con el objetivo de brindar al usuario un mecanismo que facilite la toma de decisiones, se permite seleccionar un servicio de interés como “favorito”. Es posible acceder al listado de servicios “favoritos” utilizando la opción “Ver favoritos” disponible en el encabezado de la página principal. El usuario puede seleccionar un servicio como favorito haciendo clic en el corazón a la derecha del nombre del mismo.


Figura 6: Agregar favoritos

Es posible también, eliminar elementos de la lista de favoritos definida. Esto puede realizarse haciendo clic sobre el ícono de papelera ubicado también a la derecha del nombre del servicio dentro de la lista de favoritos.

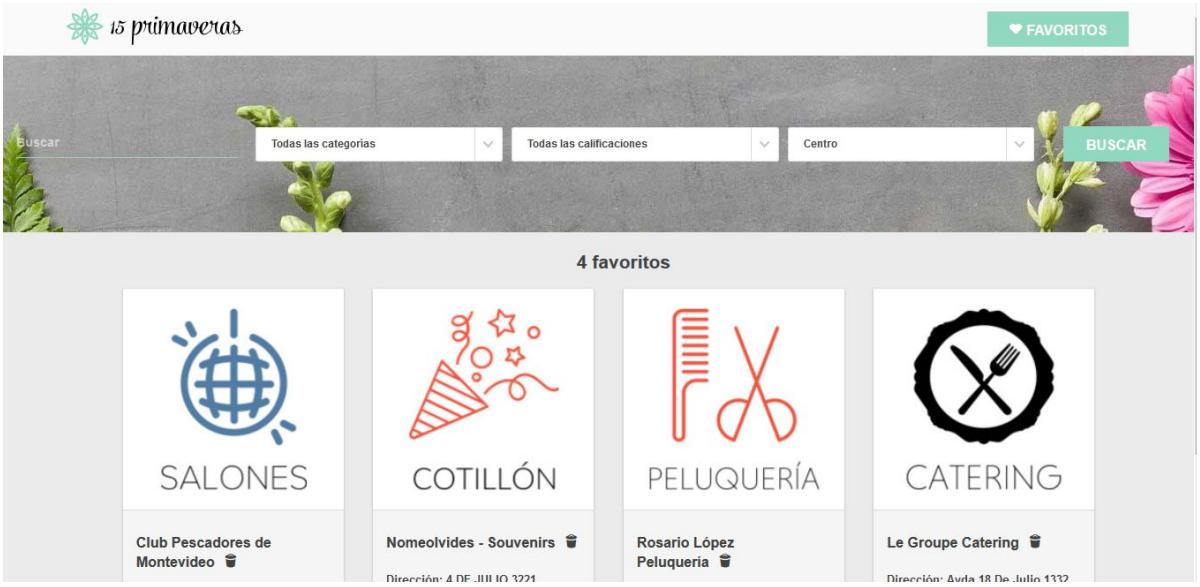


Figura 7: Ver y eliminar favoritos

### Diseño

Además de las funcionalidades mencionadas, otro de los objetivos planteados fue el diseño del sitio. Con las funcionalidades implementadas los usuarios pueden filtrar sus búsquedas, seleccionar sus preferencias y obtener información de servicios basados en calificaciones de usuarios anteriores. De todos modos, fundamentalmente por el público objetivo del sistema implementado se buscó un diseño que resulte atractivo a mujeres adolescentes.

## Evaluación y resultados

Los objetivos principales del sistema implementado fueron, aparte de brindar de forma centralizada información de servicios a contratar a la hora de organizar fiestas de 15 años, proveer de un buscador y filtros que faciliten la obtención de información de tales servicios. Fue posible implementar las funcionalidades mencionadas utilizando ElasticSearch obteniendo tiempos de respuesta extremadamente rápidos de forma tal que resulta imperceptible para un usuario.

## Conclusiones

Para la implementación del sistema fue necesario realizar una investigación de la información disponible en la web acerca de servicios a ser contratados para la organización de una fiesta de 15 años.

Luego de identificadas las fuentes de datos, se realizó un proceso de ETL en el cual fue posible no sólo aprender acerca del proceso de extracción, transformación y carga en sí mismo sino también aprender acerca las tecnologías disponibles para realizar las tareas implicadas. Si bien las herramientas disponibles facilitan el proceso y permiten automatizar un trabajo que, en caso de realizarlo manual resultaría inviable, es necesario en algunos

casos realizar transformaciones manuales. En particular, las transformaciones manuales necesarias se encuentran fuertemente ligadas a cuestiones de la semántica de los datos que las herramientas disponibles no pueden detectar.

Uno de los aspectos identificados acerca de los datos utilizados para desarrollar el sistema es el poco dinamismo que estos presentan; es decir, no son datos que se modifiquen en períodos cortos de tiempo (segundos, horas, días) sino que es posible que no varíen por períodos de meses incluso años. Es por estas razones que el hecho que la transformación de los datos haya implicado trabajo manual no presentó mayores dificultades y no es necesario contar con un proceso dedicado a extraer los datos constantemente.

Por otra parte, se valora la utilización de ElasticSearch de forma muy positiva tanto como motor de búsqueda como base de datos. Esta herramienta provee servicios para realizar las búsquedas y filtros de los datos almacenados a través de una API. Esto facilita la búsqueda de datos, en especial si lo comparamos con una búsqueda tradicional en una base de datos relacional. ElasticSearch proporciona rapidez de búsqueda y la posibilidad de escalar la solución fácilmente.

Se puede concluir que fue posible realizar una aplicación para el problema planteado utilizando las herramientas disponibles para extracción de datos y búsquedas sobre los mismos. Además, se implementó en la aplicación la posibilidad de guardar y visualizar los servicios indicados como favoritos. Se logró disponibilizar a los usuarios interesados en la organización de fiestas de 15 años un sitio en dónde pueden encontrar, de forma centralizada información sobre los servicios a contratar. Como se detalla en la siguiente sección, se proponen algunos aspectos a desarrollar como trabajo futuro.

## Trabajo futuro

Como trabajo futuro se identifican los siguientes aspectos a ser mejorados o incorporados en futuras versiones del sistema:

- Si bien en el sistema implementado se maneja el concepto de servicios favoritos, este subconjunto no se encuentra asociado a una sesión por lo cual cuando el usuario deja de utilizar el sistema la lista se pierde. Se propone para una futuras versiones la inclusión de un mecanismo de autenticación.
- La aplicación cuenta con información de servicios extraída de dos sitios web a través de procesos automatizables de dexi.io. Se propone para futuras versiones agregar otros sitios web a la extracción de datos para incorporarlos a la aplicación.
- Los datos extraídos de cada servicio proveen de información básica para que los usuarios puedan contactarse con los diferentes proveedores. De todos modos sería útil contar con más información, por ejemplo una dirección de correo electrónico o sitio web así como también imágenes asociadas a cada servicio.
- Si bien en el sistema existen servicios que no son de la ciudad de Montevideo los filtros disponibles por barrio si lo son, por ser la cantidad de servicios disponibles en Montevideo la mayoría dentro del conjunto de datos obtenidos. Se propone para

futuras versiones incluir datos de servicios en los diferentes departamentos de Uruguay para ser aplicados sobre los filtros.

## Referencias bibliográficas

[1] Miss15 Uruguay. Accesible en: http://miss15.com.uy/2016/10/ . Último acceso: Setiembre 2017.
[2] Servicios Mercado Libre. Accesible en: https://servicios.mercadolibre.com.uy . Último acceso: Setiembre 2017.
[3] El Gallito Luis. Accesible en: http://www.gallito.com.uy Último acceso: Setiembre 2017.
[4] Páginas Amarrillas. Accesible en: https://www.paginasamarillas.com.uy . Último acceso: Setiembre 2017.
[5] Uruguay Total. Accesible en: http://www.uruguaytotal.com/ . Último acceso: Setiembre 2017.
[6] TuFiesta. Accessible en: https://www.tufiesta.com.uy/ . Último acceso: Setiembre 2017.
[7] dexi.io. Accesible en: https://dexi.io/ . Último acceso: Setiembre 2017.
[8] Free formatter. Accesible en: https://www.freeformatter.com/csv-to-json-converter.html .Último acceso: Noviembre 2017.
[9] Apache Tomcat. Accesible en: http://tomcat.apache.org/ y https://hub.docker.com/_/tomcat/. Último acceso: Octubre 2017.
[10] elasticsearch.js . Accesible en: https://www.elastic.co/guide/en/elasticsearch/client/javascript-api/current/index.html . Último acceso: Noviembre 2017.
[11] Leaf. Accesible en: http://getleaf.com/. Último acceso: Noviembre 2017.