Proyecto WEBIR 2016

Grupo 4

Integrantes

Martín Steglich	4.607.084-9	msteglichc@gmail.com
Emiliano Vázquez	4.201.206-1	vazquez.emiliano@gmail.com
Matias Landeira	4.434.147-8	matilandeira@gmail.com
Cristian Bauza	4.529.320-4	cristian.bauza.silva@gmail.com

Introducción

Dada la inmensidad de noticias deportivas que existen en la red en distintos sitios, pensamos que sería adecuado contar con un sitio centralizado donde se pudieran visualizar de forma unificada las noticias deportivas de ligas, equipos y partidos de forma de facilitar a los usuarios la información deseada. Para lograr dicho objetivo se plantea usar scraping sobre ciertas páginas deportivas, almacenarlas y mostrarlas en el sitio. Se plantea un desafío en cuanto a la recolección de datos ya que cada sitio de noticias en particular tiene su formato por lo cual es necesario un trabajo en específico por cada uno.

Descripción

El trabajo consistirá en centralizar los datos relevantes y noticias de los partidos de fútbol de las principales ligas europeas.
La idea es facilitar a los usuarios de la aplicación que deseen consumir información sobre los partidos que se van a llevar a cabo en las próximas dos fechas de la liga española e italiana. De manera que el usuario ingresa el cuadro de fútbol en el que está interesado y la aplicación irá a buscar a distintas fuentes de datos, así como el diario “marca” o “la gazzetta dello sport”, e integrará la información recogida y se mostrará de manera comprensible para el usuario.
La aplicación también consumirá datos de una Api (Football Api) que tiene información estadísticas de los distintos equipos, alineaciones, horarios de partidos, etc, para lograr nutrir más los resultados de la búsqueda y de esta manera evitar que el usuario deba navegar por distintos sitios de la web para recopilar toda esta información manualmente.
Además toda la información recogida será refrescada cada treinta minutos, para proveer siempre datos actualizados. De esta manera se logra dar información fiable y actual al usuario final, ya que todos los datos consumidos son de fuentes reconocidas.

Funcionalidades

Motor de búsqueda

Nuestra propuesta incluirá integración con un motor de búsqueda, dicha herramienta deberá contar con al menos 3 de las siguientes prestaciones:

- Sugerencia ante pocos resultados, ofreciendo del vocabulario disponible aquella palabra que tenga menor distancia.
  Con esta funcionalidad colaboramos con el usuario ante errores tipográficos.
- Autocompletado, a medida que se escribe el patrón de búsqueda se genera una lista temporal que se actualiza a medida que se escribe cada carácter. Dicha lista estará ordenada por relevancia, es decir por la cantidad de apariciones de dicho patrón en las noticias.
- Orden de los resultados según relevancia, al igual que el punto anterior pero considerando las noticias más actuales y más accedidas.

- Alta performance, cada noticia presentada en este producto pasará por un proceso de tokenización según lo dictado en el curso, logrando una rápida selección de aquellas noticias que contienen dicho patrón de búsqueda.

La tecnología tentativa para este punto es Elasticsearch

La tecnología utilizada resultó ser Typeahead de AngularJS-Bootstrap

typeahead.js
a fast and fully-featured autocomplete library

Typeahead de Bootstrap es una librería bastante más reducida pero adecuada para el software desarrollado, el cual tiene una necesidad mínima de búsqueda sobre su contenido.

Web crawler

De forma metódica y automatizada, se recolectará información disponible en la Web. Así luego este contenido será indexado por el motor de búsqueda mencionado en el punto 1.

La tecnología utilizada para este punto es Scrapy

Scrapy

Integración con APIs externas

Dado el poco tiempo de desarrollo disponible y con intención de dar mayor alcance al producto, se utilizará la información que disponibilizan APIs de fútbol de acceso público.

*La API utilizada para este punto es **Football-API***

football API
affordable live score feed for developers

Tecnologías & Arquitectura

Almacenamiento de información de las APIs externas

Dado que la respuesta de las APIs es un elemento de JSON, creemos que lo más conveniente es utilizar una base de datos no relacional que tenga esta estructura de datos.

*El motor de base de datos utilizado para este punto es **mongoDB***

mongoDB®

Implementación del backend

Para la implementación del backend de la aplicación se tiene la idea de utilizar python. El mismo expondrá las funcionalidades necesarias para acceder a la información a través de una API Rest.

*La tecnología utilizada para el backend es **python***

python™

Implementación del frontend

Para la implementación del frontend de la aplicación se piensa utilizar AngularJS en su versión 2.0. La elección de esta tecnología se debe a su facilidad y rapidez de implementación. Los datos a mostrarse se obtendrán a partir de la API provista por el backend.

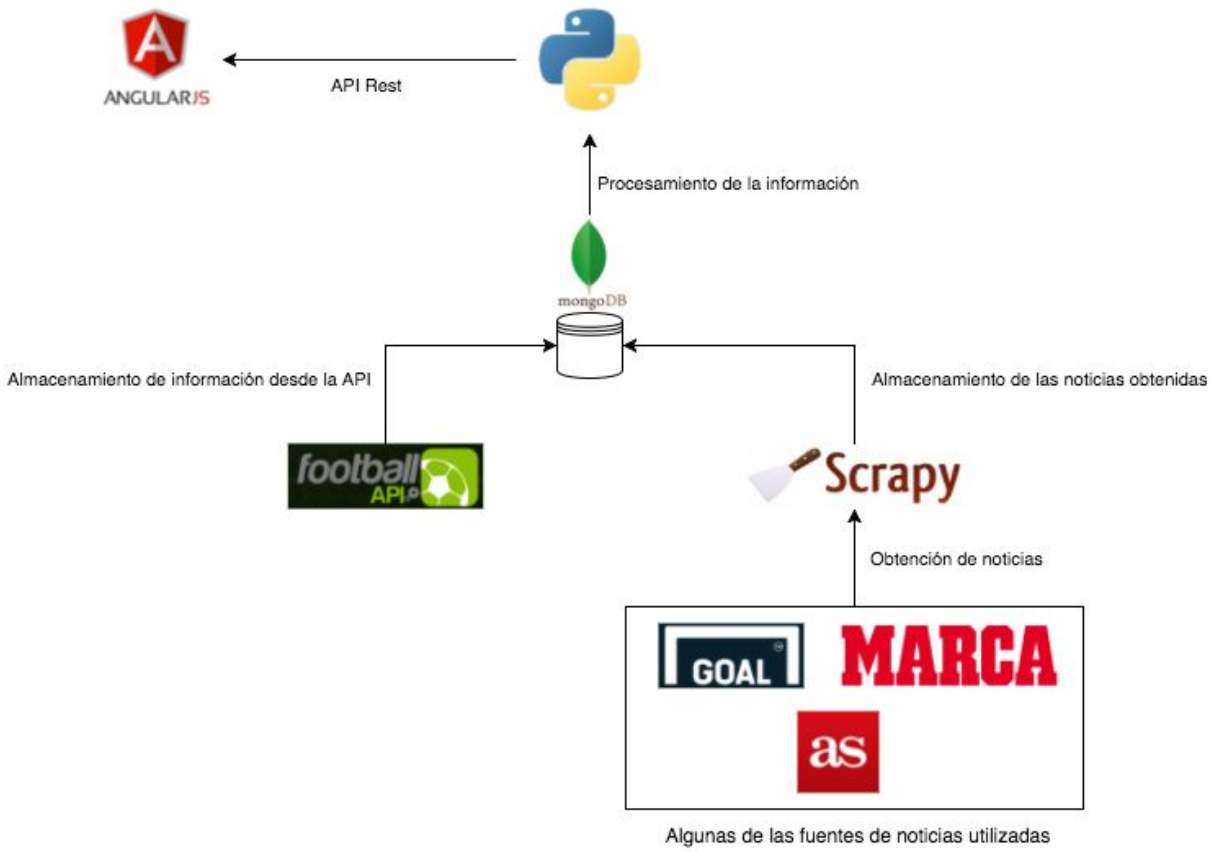
*La tecnología utilizada para el frontend es **AngularJS***

Ambiente de desarrollo

El proyecto Docker ofrece herramientas de más alto nivel que se construyen en la parte superior de algunas de las características del kernel de Linux. El objetivo es ayudar a los desarrolladores y administradores de sistemas a presentar aplicaciones y conseguir que se ejecute a través de sistemas y máquinas. Docker logra esto mediante la creación segura de LXC (contenedores de Linux) basados en entornos para aplicaciones llamadas contenedores de Docker. Estos contenedores se crean usando imágenes de Docker, que se pueden construir, ya sea mediante la ejecución de comandos manualmente o automáticamente a través Dockerfile.

Arquitectura

La arquitectura de la aplicación se basará en el siguiente diagrama, pudiendo ser modificada a la hora de la implementación.



Algunas de las fuentes de noticias utilizadas

Los datos obtenidos tanto desde la API ***Football-API*** como las noticias obtenidas desde las diferentes fuentes de noticias utilizadas a través de ***Scrapy*** serán almacenadas en una base de datos no relacional, preferentemente ***mongoDB***. La información almacenada allí, será procesada por un módulo realizado en ***python*** el cual expondrá una API Rest que será consumida por el frontend de la aplicación que estará desarrollado en ***AngularJS***.

Conclusiones

Se concluye que el stack tecnológico elegido es de muy fácil uso, permitiendo a nuestro equipo desarrollar una aplicación de mediano porte con funcionalidades interesantes en poco más de un mes.
Cabe destacar que ninguno de los integrantes del equipo contaba con experiencia previa en dichas tecnologías y que a pesar de la incertidumbre en tanto al alcance y viabilidad del proyecto, logramos cumplir en mayor parte los objetivos propuestos con un producto de calidad.

Entendemos entonces que para un desarrollo ágil el acierto estuvo en la elección de las tecnologías más adecuadas entre las disponibles para las funcionalidades propuestas, por más adecuadas nos referimos a todas aquellas que cuentan con cierta trayectoria y comunidad de usuarios detrás.
Python, AngularJS, MongoDB, Scrapy, typeahead (AngularJS-Bootstrap), son todos referentes en sus áreas.

Otro punto clave fue la utilización de una API externa dedicada, la reutilización del trabajo de otros desarrolladores de software nos dio la posibilidad de extender el alcance aún más en el plazo mencionado.

Como dificultades encontradas durante el desarrollo, destacamos que algunas de las páginas a las que hacemos Web Crawler implementan un mecanismo de protección de información, por el cual la mayoría de las veces que la accedemos automáticamente nos responde con un error de código 403 (estado HTTP Forbidden).

Trabajo a futuro

En lo que volúmenes de información refiere, en productos como el presentado en este documento se requiere una solución de espacio de almacenamiento restrictivo para la mayoría de los desarrolladores, como alternativa surge el diseño de un criterio inteligente para descartar noticias. Para esta última opción será necesario reconocer trending topics, conservando las noticias correspondientes en la base de datos. Así la mayoría de los usuarios obtendrán una lista de resultados casi al instante, mientras que para búsquedas de temas viejos o extraordinarios, nuestro producto podrá seguir dando resultados con la penalización de tiempo que significa hacer el Web Crawler a demanda.

Referencias

http://football-api.com/
https://scrapy.org/
https://angular-ui.github.io/bootstrap/

Fuentes de noticias utilizadas

http://www.marca.com/
http://www.gazzetta.it/
http://www.mirror.co.uk/