Recuperación de Información y Recomendaciones en la Web 2015

Grupo 18

Mauricio Vignale CI: 4.051.844-9
Horacio Zamora CI: 3.865.176-8
Marcelo Piñeiro CI: 4.500.070-6

Índice

Introducción

El siguiente documento describe la propuesta y solución por parte de nuestro grupo, al planteamiento de mejorar la experiencia para leer noticias en internet.

Describiremos qué componentes forman parte de la solución, mencionando que es lo que realiza cada uno de ellos, dando un ejemplo de los resultados obtenidos y por último enumerar algunos de los trabajos a futuros que se podrían realizar.

En primer lugar se explicará brevemente de qué se trata nuestro trabajo, describiendo la propuesta de mejora para leer noticias en internet, su implementación y desarrollo.
Mostraremos brevemente el resultado de usar la aplicación y una guia de usuario.

También podemos decir como introducción, dado que no había restricciones en que tecnología que se podía realizar la tarea, optamos en investigar y emplear frameworks que no teníamos experiencia, con el fin de aprender tecnologías nuevas y de ese modo no solo obtener los conocimientos que concierne a la materia en sí, sino también quedarnos con una experiencia más enriquecedora una vez realizado el curso.

Implementación y desarrollo

Propuesta

La propuesta está dirigida a aquellas personas que les gusta leer noticias en internet y que no solo tienen un sitio favorito, sino que navegan por diferentes portales e incluso accediendo a sitios de noticias que pertenecen a otros países o son internacionales.

A estas personas, seguro que en algún momento se plantearon algunas interrogantes como, ¿a qué fuente ir cuando se quiere estar informado? Si quiero navegar en más de uno, ¿existe un sitio que los centralice? ¿Puedo evitar recurrir a las mismas noticias?

Creemos que no existen, o al menos no son muy difundidos sitios en donde se encuentren centralizadas las noticias, de diferentes portales, y además que cuente con funcionalidades tales como que el usuario pueda filtrar por las noticias que le interesen y no tener que navegar por cada portal buscando las noticias, entre otras.

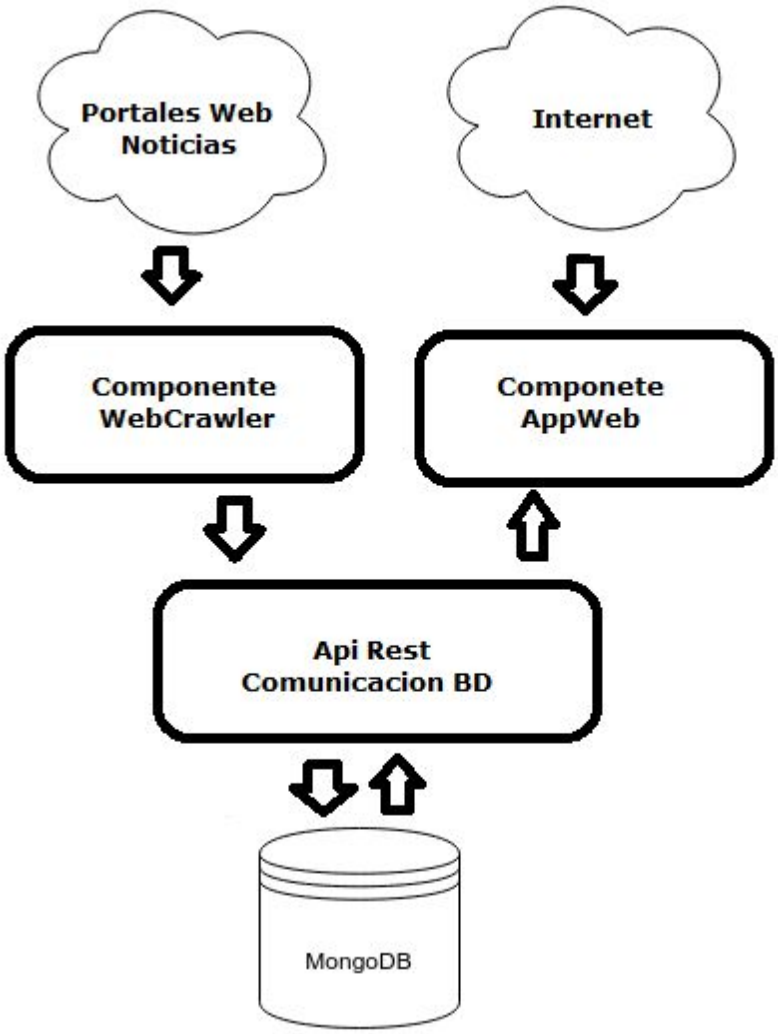
Por este motivo, propusimos la idea de presentar un sitio que centralice las noticias de los diferentes portales en un único sitio web, en donde el usuario dispondrá de distintos filtros para obtener únicamente las noticias que quiere. A su vez se contará con funcionalidades para identificar las noticias que se repitan entre los diferentes sitios, permitiendo ir a la fuente que usted prefiera.

Enfoque

Nuestra solución se basará en 2 aplicaciones: un web-crawler que recorra los diferentes portales obteniendo las noticias que se encuentran publicadas para sitios de noticias previamente establecidos, persistiendo los datos recabados en la base de datos, y la otra aplicación es un sitio web que se comunica con la misma base de noticias y le permite al usuario poder realizar algunos filtros para obtener noticias de su interés.

Estos dos grandes componentes tendrán uno en común, este componente brinda las funcionalidades de una API Rest, lo que permitirá que se comunique con la base de datos, por un lado para persistir y analizar la información, y por otro, solo para recuperar las noticias y mostrarlas, de forma desacoplada y descentralizada.

A continuación una imagen de la arquitectura:



Como Base de Datos, utilizaremos MongoDB, el cual consideramos adecuado para persistir información que no tendrá demasiada relación entre ella, haciendo a este motor ideal para esta tarea.

Para el web-crawler usaremos Node.js como servidor de aplicaciones, junto con plugins que tiene la plataforma (simplecrawler, phantom, request, entre otros), esencialmente será una aplicación por consola scrapeando los datos que obtenga de las páginas visitadas, la aplicación funciona como un web-browser headless que accede a una URL de un sitio de noticias, y desde ahí accede a todas las noticias que estén publicadas en su “home”, luego para cada noticia se realizan una serie de chequeos que determinan si se puede scrapear su contenido para luego encapsular su información y persistir en la base Mongo.

Los test de performance que hemos hecho nos han dado valores muy buenos desde el punto de vista del almacenamiento, ya que sin hacer grandes ajustes de concurrencia estamos en condiciones de visitar y

guardar cientos de páginas de diversos sitios, scrapearlas, y guardarlas sin que se afecte alguno de los pasos involucrados.

Por último para la Aplicación web, utilizaremos Angular.js, logrando así abarcar todas las tecnologías que componen el stack de aplicación MEAN (MongoDB, Express, Angular y Node.js), MEAN propone framework de aplicación Model-View-Controller con agregados interesantes que permiten una navegación del tipo "Single-Page" por lo que las acciones que se disparen de la aplicación no requiere un refresh entero de la página sino que se disparan mecanismos asíncronos de obtención y despliegue de datos (similar a AJAX de hace unos años)

La API REST para este prototipo formó parte de un módulo integrado a la Aplicación Web, con un controlador especial que indica que en la url */noticias* se accede a la API de consulta a la Base de datos de noticias, que acepta además los siguientes filtros:

- filtro por Sitio de noticias
- filtro por palabra contenida en la noticia/resumen/texto
- filtro por rango de fecha de publicación

Todos estos filtros se pasan como parámetro en la URL de la API, por ejemplo llamadas válidas a la API serian como las que siguen:

- */noticias?sitio=ElPais* (busca las noticias del diario El Pais)
- */notcias?sitio=Republica&palabra="Enrique Yanuzzi"* (busca las noticias del diario La República que contengan las palabras **Enrique** y **Yanuzzi**, no necesariamente en orden)
- *noticias?fechaDesde=2015-01-01&fechaHasta=2015-11-01* (busca las noticias de cualquier sitio publicadas de el 1ro de Enero del 2015 hasta el 1ro de Noviembre de 2015)

Pruebas

Para que nuestro sitio contenga noticias, primero hay que correr el crawler y dejarlo levantado siempre, ya que cada cierto tiempo vuelve a scrapear los sitios de noticias en búsqueda de nuevas noticias, si existen nuevas, las guarda en la base sino las ignora.

Nosotros sólo recopilamos noticias solo de algunos días, lo cual sirve de ejemplo para apreciar el aspecto pero la cantidad de noticias se vería afectado.

A continuación, una imagen del sitio centralizador de noticias, en la cual se aprecia que existe una barra horizontal donde podemos aplicar algunos filtros, entre ellos, obtener noticias a partir de un rango de fechas o buscar una palabra en su resumen o título.

Por defecto, muestra todos los portales previamente cargados, los cuales son 4, 3 portales locales y uno internacional (Argentina), pero se permite ver de un solo portal.

El sitio mostrará la cantidad de noticias recuperadas, incluyendo si se aplica algún filtro. Bajo un panel desplegable por el título del portal, se agruparan las noticias de dicho portal, viéndose el título de la noticia, el resumen, la fecha de esta y un botón (Ver noticia >>) para ir a la fuente de la noticia en caso de querer ver desarrollada la noticia.

Se puede ver que existe un filtro por palabras, el cual en caso de ingresar más de una separadas por un espacio, busca estas palabras dentro del título y resumen, verificando que existan todas las ingresadas

Trabajos futuros

Como vemos hay un boton que dice “Noticias similares”, la idea era poder detectar noticias que con algunos cambios en el título o la redacción pertenecieran a la misma noticia y así darle la posibilidad al lector de elegir la fuente.

Esta funcionalidad había sido nuestro gran reto, pero no logramos realizarla ya que no nos resultó un problema trivial y no pudimos encontrar alguna librería que nos resolviera el problema.

Si bien, nos habían sugerido la librería “Solr”, investigamos, realizamos algunas pruebas pero el problema en concreto no lo solucionamos, aunque somos optimistas que se puede hacer.

Por otro lado, existen pequeñas mejoras que se podrían hacer para aumentar la experiencia del usuario y algunas de ellas son:

- Poder agregar sitios dinámicamente y no que solo podamos leer de aquellos previamente cargados, si bien hay que tener un tiempo para procesar las noticias al agregar un sitio nuevo, se podría dejar en estado “procesando”, y una vez listo dejarlo disponible, ademas quedaria para próximos lectores.

- Limitar la cantidad de noticias a ver por defecto, por ahora trae todas las que se se obtuvieron con la aplicación.

- Que al buscar palabras en las noticias, busque también dentro del cuerpo de estas, por ahora solo busca en el título y resumen obtenido, pero al no persistir todo el desarrollo de la noticia no podemos procesarla. Si bien no parecía difícil de realizar pero invertimos el tiempo en otras cosas que creímos más importante.

- Programar el “crawler” para obtener noticias antiguas y de esa forma quedarnos con una especie de historial de noticias, dado que las noticias que recoge son solo desde que empieza a correr la aplicacion en adelante.

- Si se pudiera categorizar las noticias se podrían agrupar a estas para que elija las de su interés, por ejemplo, si me interesa mucho

las noticias de fútbol, y contamos con una sección de Deportes, agrupar en este todas las noticias de fútbol.

- Modificar el código para obtener las noticias de forma genérica y no personalizarlas según el sitio, vimos que cada sitio tiene su estructura para maquetar las noticias y eso nos resultó un poco engorroso a la hora de hacer soluciones genéricas.

Conclusiones

Con respecto a elegir nuevas tecnologías para realizar la tarea fue una opción muy buena, nos resultó mucho más fácil poder desarrollar las aplicaciones ya que eran librerías muy potentes, livianas, y en JavaScript lo cual alguna vez habíamos visto, lo que nos facilitó para su aprendizaje.

Por el lado de la aplicación, vimos que el resultado era muy bueno, mejoraba la experiencia para leer noticias en internet, pudiendo ir a las noticias de interés aplicando los filtros, y no había que estar navegando entre los diferentes portales.

Además, al tener el resumen de la noticia sin publicidad, y sin imágenes, podemos concentrarnos en lo que dice de esta y así evitamos el ruido visual que algunas nos resulta demasiado molesto.

Manual de usuario

Para tener las aplicaciones andando, hay que realizar una serie de pasos muy fáciles que pasamos a detallar.

Dado que estamos utilizando Nodejs como librería para la lógica, esta debe estar previamente instalada en la máquina que se realizarán las pruebas, es pre-requisito para que funcione todo. La referencia [1] explica cómo hacerlo. Teniendo esto resuelto continuamos.

En primer lugar, en la máquina que se va a correr las aplicaciones debe estar instalado y levantado un motor de bases de datos, en nuestro caso optamos por usar MongoDB.
Vamos a donde quedo instalado y ejecutamos el servicio mongod.exe, en caso de estar corriendo en Windows.

…….\MongoDB\bin>mongod.exe

Esto levanta un intérprete de comandos que nos deja realizar querys contra el Mongo, lo único que debemos hacer para dejar habilitada la inserción de datos de noticias es ejecutar la siguiente instrucción

#>db.noticias.createIndex({ "$": "text" })**

Esto crea un índice de texto sobre todos los campos de la BD noticias. como Mongo es un sistema de datos no relacional no es necesario definir un esquema estructurado de datos, ya que todo es un "documento" y no tiene porque tener formato.

Una vez corriendo el motor de BD, procedemos a correr la aplicación "Crawler" la cual obtendrá las noticias de los diferentes portales, previamente cargados a mano en el código.

Primero con una consola de windows/terminal de unix nos posicionamos sobre el directorio webir-crawler y ejecutamos:

…….\webir-project\webir-crawler>npm install

npm es un utilitario de node que nos permite declarar cuales son las dependencias de nuestro proyecto (este archivo se encuentra en el package.json) para luego poder instalarlas desde este comando que ejecutamos. cuando esto termine corremos:

…….\webir-project\webir-crawler>npm start

Con ello veremos que el sistema empieza a correr, recorriendo los sitios predefinididos en el archivo *scraping_v4.js* y almacenando los datos en la instancia de Mongo levantada (BD noticias, collection noticias).

Con esto tenemos datos suficientes como para poder poner en marcha nuestro frontend. asi que finalmente levantamos la aplicación del sitio, para ello vamos a donde está alojada nuestra aplicación y nos posicionamos en la carpeta que contiene al archivo app.js (webir-proyect\webir-webapp), una vez allí digitamos con la consola de comandos: npm start y si todo salió bien debería desplegarse un mensaje con el puerto para levantar.

#>npm start
Se creo el server express en el puerto: 3000
Conectado a MongoDB

Ahora podemos entrar a la dirección **localhost:3000** y empezar a usar la aplicación.

Referencias

Si bien las referencias que podemos dejar son solo para aprender de NodeJs y AngularJs, aquí dejamos algunos link que nos sirvieron.

[1] http://www.desarrolloweb.com/articulos/instalar-node-js.html
[2] https://www.youtube.com/watch?v=ymvfhkk8dC0
[3] https://www.codejobs.biz/es/blog/category/angular/page/4/#top