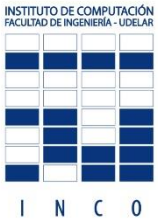


Facultad de Ingeniería, UdelaR

Recuperación de Información y Recomendaciones en la Web 2017

Surtite Ya!

Grupo 15
Ignacio Fiori, 4.940.793-8
Mateo Mujica, 4.596.060-5
Bruno Scarone, 4.706.574-0

Docente:
Libertad Tansini

Contenido

1. Introducción

Hoy en día existe una gran cantidad de supermercados para que los consumidores se abastezcan de los productos necesarios para su consumo diario. Para determinar el lugar donde el consumidor efectuará la compra, usualmente se consideran como factores relevantes: el precio de los productos a ser adquiridos, la cercanía del lugar y la marca de los mismos, entre otros. Para escoger el supermercado donde una persona puede obtener el conjunto de productos deseados al menor precio (compra óptima) es necesario recorrer varios establecimientos y realizar el registro y la comparación de los precios para los distintos productos.

En los últimos años, los supermercados han incorporado servicios de venta y envío online a través de páginas web, alternativa cada vez más popular entre los consumidores. Esta modalidad facilita la determinación de la compra óptima, ya que es posible comparar precios accediendo a los sitios de los distintos supermercados, sin necesidad de desplazarse a los lugares físicos de los mismos. Sin embargo, sigue siendo trabajoso adquirir toda la información, ya que la misma está distribuida en varios sitios web.

Dadas estas condiciones se propone brindar una interfaz unificada que centralice información de precios y variedad de productos disponibles en el mercado para facilitar la especificación de la compra óptima por parte del consumidor.

2. Definición del Problema

El problema abordado consiste, como mencionamos en la sección anterior, en construir un sistema que posibilite la determinación del supermercado donde se puede realizar una compra óptima en términos de precios basada en la especificación de una lista de productos brindada por el usuario del sistema.

3. Enfoque de la solución

Se define para este trabajo un conjunto de páginas web, correspondientes a distintos supermercados uruguayos, desde el que se obtendrá información relativa a los productos que estos comercializan. El enfoque de la solución consiste entonces, en obtener la información antes mencionada y crear una página web donde se puedan buscar distintos productos, consultar su disponibilidad en los distintos supermercados y determinar para un conjunto de productos que están incluidos en una canasta básica, el supermercado en el que la compra de los mismos resulta óptima en términos de precio.

4. Tecnologías y herramientas utilizadas

Presentamos a continuación las distintas tecnologías y herramientas utilizadas para el desarrollo de la solución propuesta.

Todos los componentes de la solución fueron desarrollados en un sistema operativo Ubuntu [1] versión 16.04. Para el desarrollo de la capa BackEnd del sistema se utilizó el lenguaje de programación Python [2] en su versión 2.7, junto con un framework web de código abierto desarrollado sobre el mismo llamado Django [3] en su versión 1.11.7. El mismo implementa el patrón arquitectónico model-view-template (MVT), utilizando un módulo que realiza el mapeo objeto-relacional (ORM) entre modelos de datos definidos por el usuario como clases en Python a bases de datos relacionales; un sistema para el procesamiento de pedidos HTTP con un sistema de templates web y haciendo uso de un despachador de URLs basado en expresiones regulares para dirigir los pedidos a las vistas correspondientes.

En lo que respecta al FrontEnd, se utilizaron tecnologías web como HTML, CSS, JavaScript para el renderizado de todos los templates y en especial del formulario donde el usuario podrá realizar la búsqueda de los productos que desee. Se utilizó también la técnica AJAX (Asynchronous JavaScript And XML) para tener una comunicación con el servidor más eficiente. Las versiones de los mismos son HTML 5, CSS3 y ECMAScript7 la versión de JavaScrpit.

La tecnología utilizada para la extracción de la información de las distintas fuentes de datos fue el framework de código abierto para Python, Scrapy [4] en su versión 1.4.0. Scrapy permite realizar web crawling, con el fin de extraer información de manera automática de distintas páginas web. Esto se realiza mediante la definición de módulos denominados como rastreadores o arañas web.

Con respecto a la persistencia de la información, se utilizó una base de datos relacional con el sistema de gestión de bases de datos relacionales (DBMS) PostgreSQL [5] en su versión 9.5.9.

5. Arquitectura y diseño de la solución

Exponemos a continuación un diagrama de la arquitectura del sistema desarrollado y explicamos las capas y los componentes de las mismas.

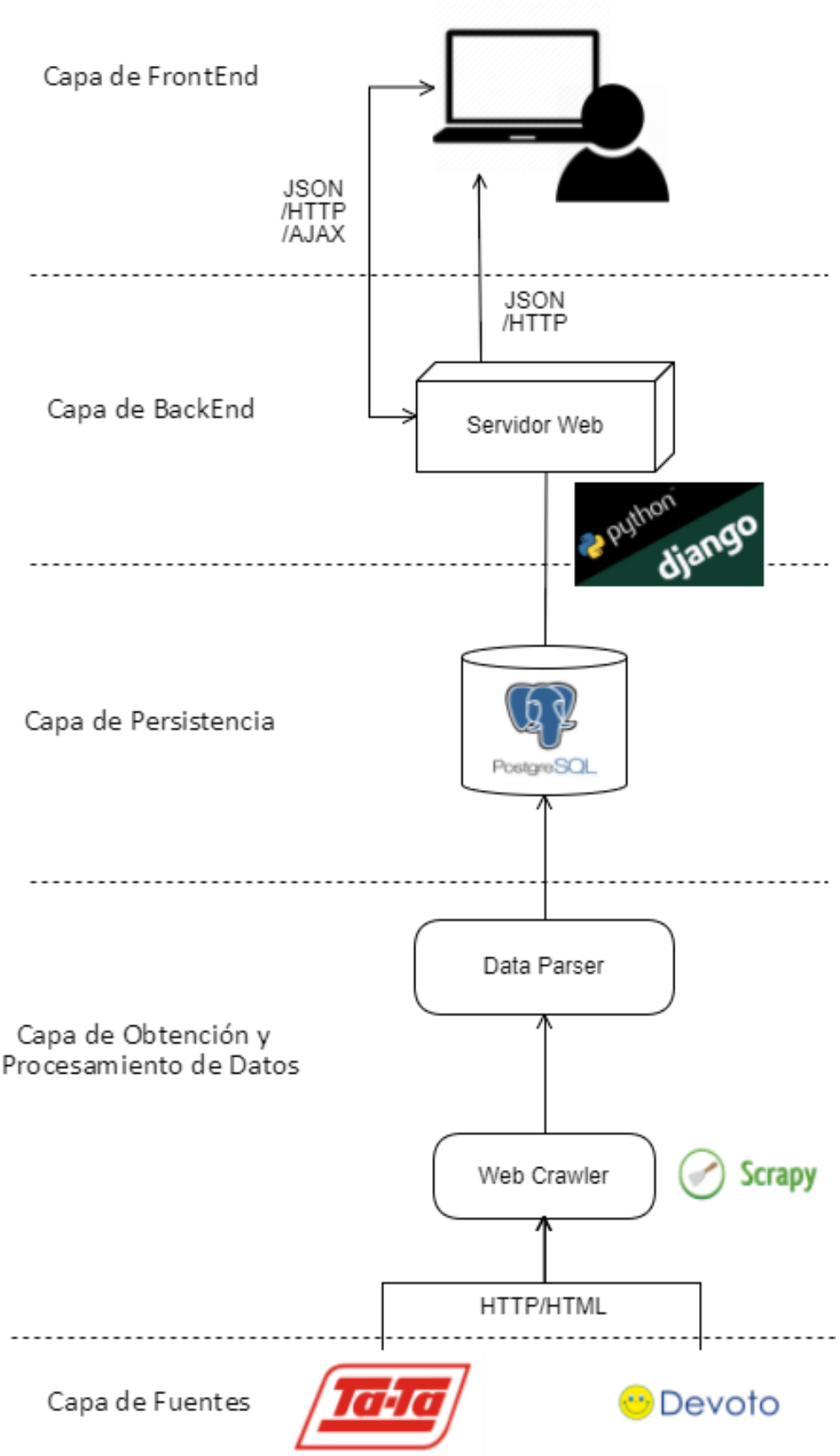


Diagrama de Arquitectura del Sistema

5.1 Capa de Fuentes

Esta capa está compuesta por las páginas web de los supermercados considerados para la obtención de datos. En la siguiente sección explicaremos las páginas elegidas y los métodos para realizar la extracción.

5.2 Capa de Obtención y Procesamiento de Datos

En esta capa se encuentran presentes las arañas (spiders) que utilizan las funcionalidades provistas por el framework Scrapy para obtener los datos de los productos disponibles en las distintas fuentes de datos.

La salida del web crawler es posteriormente ingresada en el módulo de Data Parser, donde se analizan los datos y se obtener los elementos necesarios para realizar la carga de las tablas de la base de datos correspondiente. Además, en este módulo se realizaron tareas de normalización tanto para la dimensión sintáctica de los datos como para las dimensiones de las medidas utilizadas para la cuantificación de las cantidades de los productos. Profundizamos estos aspectos en la sección de implementación de esta capa.

5.3 Capa de Persistencia

En esta capa se encuentra el esquema relacional junto con las distintas tablas que almacenan los datos utilizados por el sistema. En una sección futura explicaremos el modelo de datos utilizado y la estructura de la base de datos donde los mismos son almacenados.

5.4 Capa de BackEnd

En esta capa se encuentra el servidor de aplicación, que recibe los pedidos de los distintos clientes web, los analiza y retorna las respuestas apropiadas para los distintos casos. Aquí es donde se definieron los modelos de datos que permiten realizar el mapeo para la persistencia y su posterior recuperación de la base de datos que fue cargada con los productos obtenidos en la capa de Obtención y Procesamiento de Datos.

5.5 Capa de FrontEnd

En esta capa del sistema se da la interacción del mismo con el cliente, utilizando el último la interfaz web diseñada. El navegador en la máquina utilizada por el mismo realizará

pedidos al servidor web y utilizando la información obtenida mostrará respuestas al cliente para que pueda así hacer uso del sistema.

6. Fuentes de datos utilizadas

Al momento de la realización de este trabajo, se constató que únicamente tres supermercados uruguayos ofrecen sus productos a través de sus páginas web, estos son Tienda Inglesa [6], Devoto [7] y Tata [8]. Es esto por lo que, al comenzar el análisis y la definición de las fuentes de datos a utilizar, se definió considerar las tres páginas web para obtener los datos que usaría el sistema.

Sin embargo, se decidió posteriormente descartar como fuente de datos los productos del supermercado Tienda Inglesa, por los motivos que detallamos a continuación. Observando la página y la dinámica de mensajes que la misma intercambia con el servidor que contiene la información de la página a través del navegador se estableció que:

El sitio utiliza scrolling infinito (Infinite Scrolling) para cargar los distintos productos, comenzando con 8 productos cargados inicialmente cuando se accede a la página de una categoría y luego a medida que se realiza el scrolling se van generando pedidos al servidor que incorporan hasta 8 nuevas unidades por pedido.

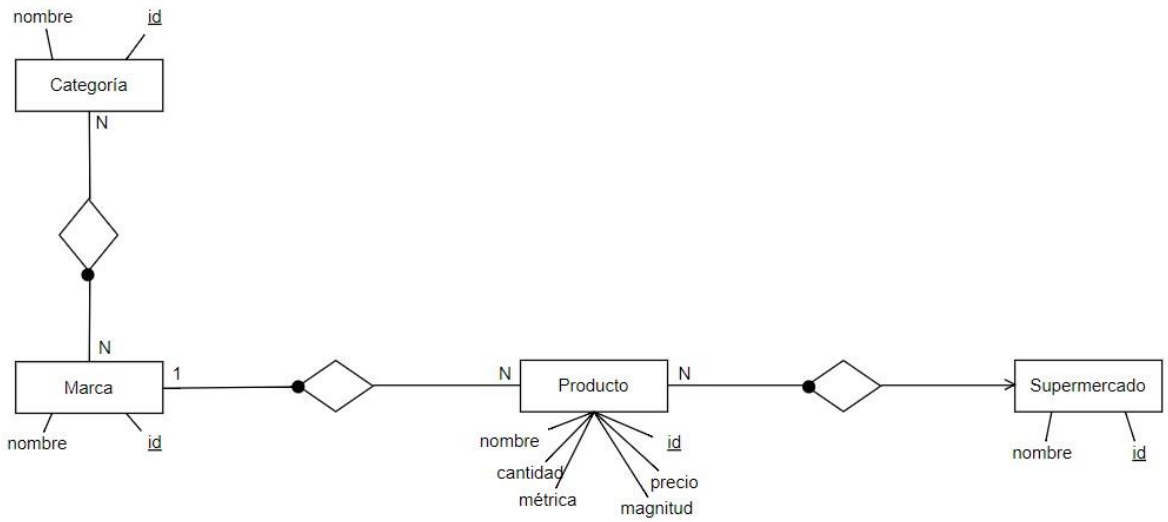
Se investigaron distintos métodos para obtener la totalidad de los productos antes o durante la realización del scrapeo de los mismos. Como no se puede obtener información de los distintos pedidos a través de ningún elemento visible en la página, como una url, esto complicó la realización de esta tarea. Además, utilizando un inspector de red se observó que los distintos pedidos incluían dos tokens de autenticación cuyo método de generación era desconocido. Finalmente, se logró simular pedidos basados en información extraída de otros observados por el inspector web en el navegador. Esta simulación puede darse de forma efectiva por un periodo de tiempo determinado, perdiendo los tokens validez luego de este y por consiguiente el pedido.

Debido a la cantidad de productos que se desean obtener y la cantidad que son obtenidos por pedido, no se considera que el método sea suficientemente eficiente para ser utilizado, además de que el mismo no puede ser reproducido de manera automática una vez pasado el periodo de tiempo de validez de los tokens del mismo. Además, una vez comenzado el análisis se evaluó que la complejidad y los desafíos de lograr la implementación del prototipo con dos fuentes ya eran relativamente altas y en particular dos fuentes resultan suficientes para la realización de una prueba de concepto de las funcionalidades a implementar.

El sistema implementado contiene entonces la información relativa a los productos de la canasta básica disponibles en los supermercados Devoto [7] y Tata [8].

7. Modelo de Datos del sistema

Presentamos a continuación el modelo de datos utilizado en el sistema. El mismo viene dado por el siguiente modelo entidad relación:



Modelo Entidad Relación de los datos del sistema

Seguimos con la descripción del mismo. La entidad central de nuestro modelo de datos y nuestro sistema son los productos, que están identificados por un id que tomará la forma de una clave subrogada numérica entera asignada en el momento de la creación del mismo y el identificador del supermercado en el que este es ofrecido. Luego los productos cuentan con los atributos nombre, cantidad en caso de que este sea ofrecido como un pack de múltiples unidades, el precio y la métrica y magnitud si corresponden.

Luego, en nuestra realidad se cumple que todo producto es parte de un supermercado que lo comercializa y por tanto se modela el producto como una entidad débil del supermercado en el diagrama ER. Esto pone de manifiesto lo explicado con anterioridad respecto a la pareja de atributos que identifica un producto: su identificador y el del supermercado en cuestión.

Continuando con la marca, la misma es identificada por un valor numérico entero único asignado por el sistema a la hora de su creación y contiene su nombre. Toda marca debe estar presente en al menos un producto del sistema, pudiendo existir varios productos correspondientes a la misma. En caso de que un producto cuente con una marca, la misma debe ser única.

Finalmente, las categorías también identificadas por una clave subrogada y que cuentan además con un nombre agrupan varias marcas. Toda marca debe estar en al menos una categoría, pero la misma puede estar presente en más de una.

8. Implementación

Detallamos en esta sección consideraciones relativas a la implementación de las distintas capas del sistema presentadas en la sección de arquitectura y diseño del mismo.

8.1 Capa de Obtención y Procesamiento de Datos

Como se mencionó en la sección de arquitectura, en esta capa se encuentran los módulos de Web Crawling y de Data Parsing del sistema. Profundizamos ahora en la forma en que los mismos fueron implementados. El módulo de Web Crawling está compuesto por las arañas, que hacen uso de Scrapy y toman como entrada las urls de las páginas webs de los supermercados, brindando como salida brindan un archivo en formato json que contiene todos los productos distinguidos por categoría, su nombre, junto con el precio al que son comercializados en el supermercado correspondiente. La spider realiza pedidos HTTP a la dirección de los distintos supermercados y recibe en las respuestas HTTP, páginas HTML que luego inspeccionará para extraer las secciones correspondientes donde se encuentre la información de interés, que en nuestro caso involucra obtener, para las distintas categorías disponibles vinculadas a productos de la canasta básica, la información de los mismos que detallamos anteriormente.

Luego se utilizó un módulo de parseo de datos, para realizar correcciones sobre los datos obtenidos de las distintas fuentes con respecto a codificación de algunos caracteres especiales como tildes o el carácter “ñ”. También se uniformizaron los nombres de los productos, eliminándose espacios extras presentes en algunos de los elementos al comienzo o al final del campo y también los de carriage return (\r) y line feed (\n) si estaban presentes.

Una vez finalizada la estandarización y corrección sintáctica de los datos obtenidos, se continuó con el procesamiento en el módulo de Data Parser y para cada producto disponible en el archivo json obtenido como salida del módulo de Web Crawling, se utilizaron expresiones regulares para obtener los elementos necesarios para realizar la carga de las tablas de la base de datos correspondiente. Se obtuvo entonces para cada producto los siguientes atributos: su nombre, el precio al que este es vendido en el supermercado correspondiente, la cantidad de elementos que contiene el producto considerado (si corresponde) y la magnitud junto con la métrica (si corresponden) en la que se ofrece dicho producto. Además, se realizó en este módulo la normalización de las unidades utilizadas para las cantidades en las que los productos son ofrecidos con el fin de facilitar la tarea de procesamiento en la capa de BackEnd.

Dependiendo de qué spider provenía el archivo utilizado, se determinó el supermercado al que correspondían los productos considerados. Concluida esta etapa se procede a, con los atributos obtenidos para cada producto, realizar la carga de los mismos en la

base de datos utilizada. Para esto se utiliza la interfaz provista por Django para realizar persistencia de datos.

Tanto para las categorías como para las marcas utilizadas, se construyeron diccionarios que contienen las mismas. Estos fueron cargados mediante la ejecución de dos scripts que también hacen uso de la interfaz provista por Django para la carga de datos.

8.2 Capa de Persistencia

Luego de crear una base de datos para almacenar los datos del sistema, se realizó la creación de tablas y la carga de las mismas mediante la interfaz provista por el entorno de trabajo, como mencionamos en la subsección anterior. De esta forma, fueron cargados las siguientes cantidades de productos, que exponemos en la tabla a continuación:

Supermercado	**Cantidad de Productos Cargados**
Tata	3996
Devoto	4649

Presentamos ahora la cantidad de categorías y marcas disponibles en el sistema:

Cantidad de categorías cargadas en el sistema	7
Cantidad de marcas cargadas en el sistema	322

Las categorías elegidas son las siguientes: Almacén, Bebidas, Congelados, Elaborados, Frescos, Limpieza y Lácteos.

8.3 Capa de BackEnd

Para implementar las operaciones básicas para el conjunto de objetos del sistema, compuesto por Productos, Supermercados, Marcas y Categorías, se utilizó la librería Django Rest [9]. Esto permite generar, especificando un conjunto de parámetros relativos al modelo, implementaciones para las operaciones de creación, modificación, obtención, listado y borrado para el objeto considerado. Todas estas operaciones se alinean con el estilo arquitectónico REST (Representational State Transfer) [10].

Luego, fueron necesarias implementar dos vistas adicionales para completar las funcionalidades necesarias que debían ser brindadas por el servidor de la aplicación. La primera consiste en para las distintas especificaciones de productos definidas por el usuario en la interfaz gráfica del sistema, devolver los productos que cumplen con la

misma para cada una de ellas. Cada especificación viene dada por: una categoría, una marca (ambas elegidas dentro de los diccionarios utilizados) y un nombre de producto ingresado por el usuario. Se retorna entonces una lista en formato json de los productos posibles para cada uno de los ítems especificados, con la información necesaria asociada a cada producto. Como se verá en las siguientes secciones, esto busca orientar al usuario en la elección de los productos disponibles en los supermercados considerados antes de realizar el cálculo de la compra óptima.

Finalmente, la última vista implementada realiza la operación central del sistema: para un pedido dado por el usuario, determina el lugar, así como otra información de interés relativa a la compra óptima para ese pedido. En este caso el pedido consiste en una serie de especificaciones de la forma: nombre del producto, cantidad y la magnitud tanto como la métrica en caso de que corresponda para el producto considerado. En base a estos datos, se consulta en primer lugar la disponibilidad del producto en ambos supermercados. Si el mismo solo se encuentra en un único supermercado, se verifica que las cantidades requeridas por el usuario puedan ser satisfechas con las disponibles en el supermercado donde hay disponibilidad del producto. En caso de que esto no sea así, se considera que este ítem del pedido no puede ser satisfecho. Si el producto se encuentra disponible tanto en Tata como en Devoto, se pasa a verificar que las cantidades en las que lo ofrece cada supermercado sean adecuadas para satisfacer el ítem dado. En la ejecución de la operación se registra el total de cada pedido para cada supermercado, así como la cantidad de productos no disponibles en cada uno de ellos para el pedido dado.

Se retorna entonces la respuesta en formato json que contiene dos listas que corresponden a los ítems de cada supermercado para cada elemento del pedido que lo satisface, las cantidades necesarias para el mismo y el precio correspondiente. Para determinar el lugar óptimo donde debe realizarse la compra, se considera primero el lugar con mayor cantidad de ítems disponibles como la mejor opción, sin importar el precio, priorizándose así la completitud del pedido por sobre esta variable. Luego, si la cantidad de ítems no disponibles es la misma para ambas opciones, se considera el precio para determinar el supermercado.

Observamos que la operación no se encuentra sesgada hacia ningún supermercado en los casos de borde, ya que, si ningún supermercado cuenta con los productos especificados, la respuesta será que en ninguno de los dos es posible realizar la compra. Si la cantidad de productos no disponibles en la misma en ambos casos y el precio al que se puede obtener el resto es igual, se responderá que cualquiera de las dos opciones es adecuada para cumplir el objetivo.

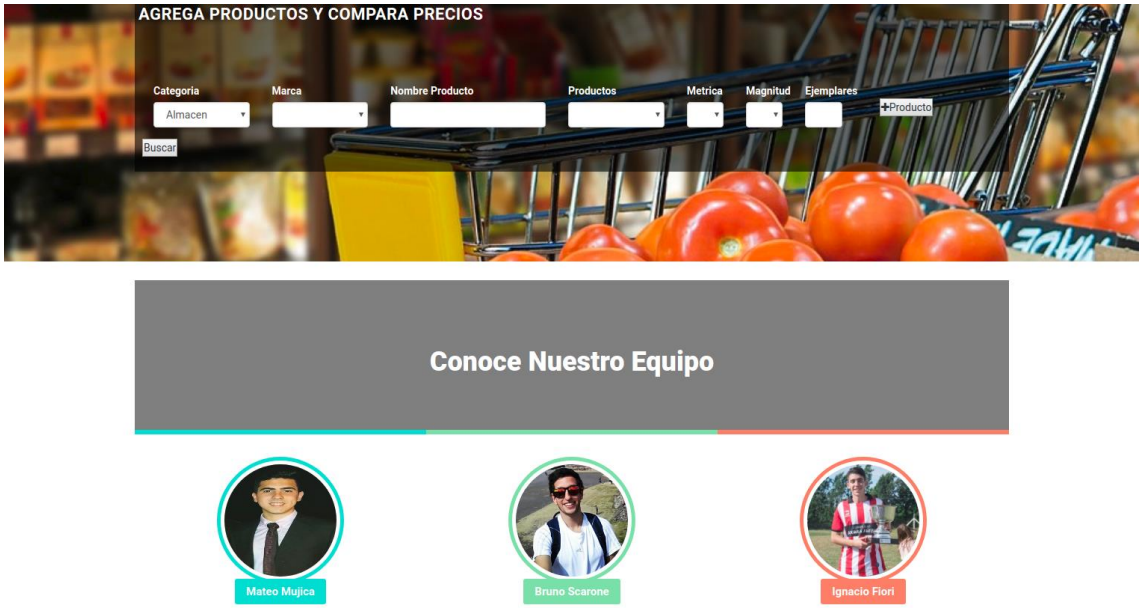
Notar que, por construcción de la interfaz provista y las operaciones diseñadas, sabemos que en todos los casos el producto existirá en al menos un supermercado, sin embargo, puede pasar que las cantidades requeridas por el usuario no puedan ser satisfechas. Esto puede originar casos en los que ninguno de los ítems del pedido pueda obtenerse en los supermercados considerados, dando lugar a la opción de respuesta de que ninguno de los dos cumple con el objetivo, que presentamos en el párrafo anterior.

8.4 Capa de FrontEnd

Presentamos en esta sección las interfaces gráficas desarrolladas para el sistema, así como una descripción de la forma en la que se utilizan las operaciones definidas en la capa de BackEnd que describimos en la subsección anterior para satisfacer los requerimientos del sistema.

Mostramos a continuación la página inicial que es cargada al acceder al sitio web desarrollado.

En esta, se cuenta con la interfaz gráfica a través de la que un cliente puede especificar los distintos productos que componen su pedido. Describimos a continuación el flujo de acciones que deben ser realizados por un cliente para generar un ítem en su pedido.



Primero, debe especificar una categoría, seleccionando un elemento de la lista desplegable que encontramos correspondiente a la categoría del producto. Las categorías son cargadas consultando el diccionario de categorías almacenado en el servidor una vez se carga la página inicial.

Luego, la elección de las categorías genera un pedido AJAX que consulta las marcas asociadas a la categoría elegida, que se encuentran almacenadas en el servidor web. Estas son cargadas en la lista desplegable del ítem correspondiente.

A continuación, el usuario ingresa en la barra de búsqueda del ítem el nombre del producto que está buscando para incluir en su pedido, por ejemplo, puede escribir el string “leche”. Una vez escrito el nombre del producto buscado, se realiza una consulta AJAX de manera dinámica al servidor que ejecuta la operación de búsqueda de producto implementada por una de las vistas que fueron comentadas en la sección anterior, brindando esta como resultado los productos cuyo nombre contiene la palabra especificada previamente. Notar que esto puede dar lugar a múltiples productos de los distintos supermercados, para el ejemplo anterior se recuperan los siguientes productos: leche entera, leche descremada, leche en polvo, dulce de leche, leche chocolatada, entre otros.

Una vez se cargan los productos ofrecidos por los distintos supermercados, el usuario elige el que desee incluir en su pedido. Esto carga las métricas y las magnitudes en los que este es ofrecido en los supermercados en los que esté disponible. El usuario continúa el proceso eligiendo una métrica y magnitud en las que desee adquirir el producto elegido. Especifica finalmente la cantidad deseada para el ítem elegido.

Este proceso puede ser repetido para tantos productos como quiera el usuario del sistema y una vez que la lista del pedido esté completa el mismo procede a presionar el botón “Buscar”.

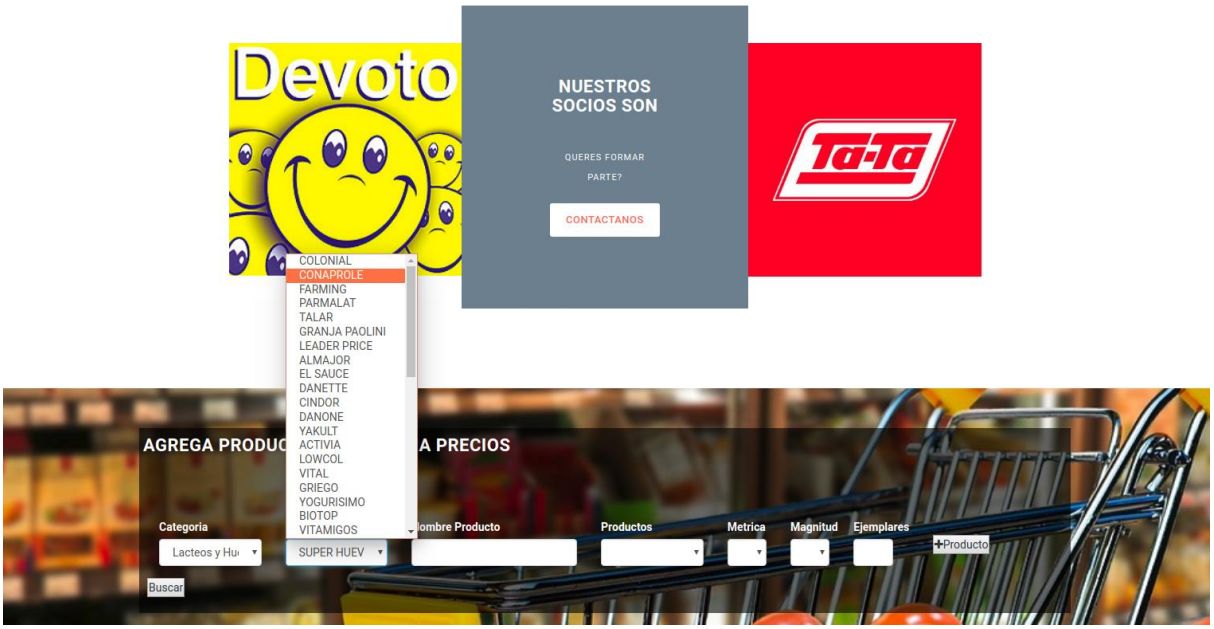
Esto genera un pedido que hace uso de la operación que determina la compra óptima, su lugar de realización (en caso de que exista) y otra información correspondiente a esta que describimos en la sección anterior. Luego, se redirige la navegación a la página que despliega el resultado haciendo uso de la información proporcionada por el servidor web.

9. Funcionalidades y uso

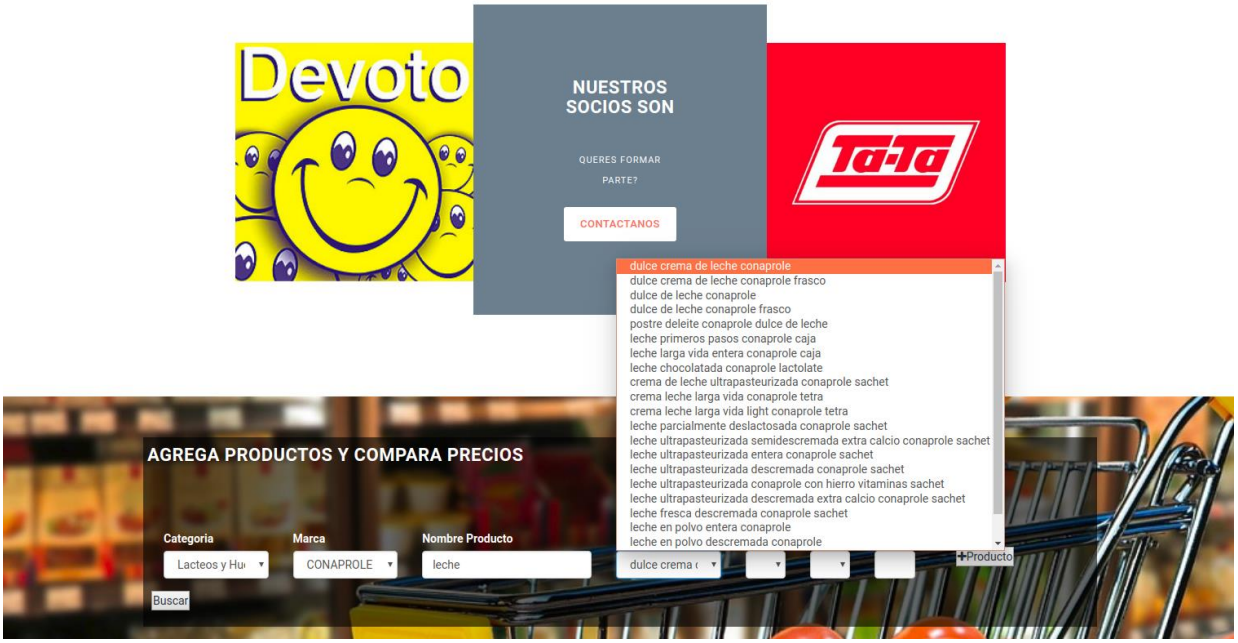
Resumimos y explicamos en detalle las funcionalidades ofrecidas por el sitio a los usuarios del mismo. Como se comentó anteriormente, existen dos funcionalidades en el sitio: la especificación de un pedido y la búsqueda de la compra óptima para el mismo.

Presentamos ahora imágenes que ilustran la especificación de un pedido por parte de un usuario utilizando las interfaces gráficas que mostramos en la sección anterior.

El flujo comienza mediante la selección de una categoría, en este caso se eligió la categoría “Lácteos y Huevos”. Como se mencionó antes, esto carga las marcas asociadas a dicha categoría como vemos en la imagen.



Una vez elegida la marca, se escribe el nombre del producto buscado. En este caso se realizó una búsqueda con el término “leche”. Esto carga de manera dinámica todos los productos ofrecidos en los supermercados, cuyo nombre contiene los términos de búsqueda.



Finalmente se especifica la métrica y la magnitud en la que se desea obtener el producto dentro de las opciones disponibles y se ingresa la cantidad de ejemplares que se desean obtener en la compra.

Este proceso puede reiterarse para un nuevo ítem a ser incluido en el pedido si se toca el botón “+Producto”. Una vez incluidos todos los productos del pedido que el usuario desea adquirir el mismo procede a presionar el botón “Buscar”.

Luego, se redirige la navegación hacia la página de resultados que mostramos en la imagen siguiente.

Producto	Magnitud	Metrica	Cantidad	Tata	Devoto
Leche Conaprole	1000	ml	2	23	23
Jugo Ades Manzana	1000	ml	1	74	79
Queso Crema Azul Farming	200	g	1	106	93
Agua Mineral Matutina Con Gas	2225	ml	2	70	72
Jamon Cocido Ottonello	1000	g	1	390	380

	Tata	Devoto
Cantidad No Disponible	0	0
Total Compra	756	742

Tu mejor opción es

Además de los precios a los que son adquiridas las distintas cantidades de los productos necesarias para satisfacer el pedido realizado y la información asociada a los productos, se muestra la cantidad total de productos no disponibles por supermercado, así como el total de la compra.

Siguiendo los criterios que se explicaron en la sección de Back End para las operaciones, se determina la mejor opción para realizar la compra especificada, siendo la imagen correspondiente a el supermercado elegido mostrada en pantalla como puede verse en la figura. Para los casos en los que cualquiera de los supermercados sea adecuado o que ninguno cumpla las condiciones pedidas, se muestran imágenes que indican lo antedicho.

10. Conclusiones

En el trabajo se explicó la metodología y las técnicas utilizadas para la construcción de un sistema que permita determinar, dada la especificación de un pedido constituido por productos de la canasta básica, el supermercado en donde puede realizarse la compra óptima del mismo en términos de precio, en caso de que esta exista.

Para esto comenzamos presentando la motivación para la construcción del sistema y la utilidad que el mismo puede brindar a los usuarios. Se definieron en base a estos objetivos tanto el problema como el enfoque de la solución que fue implementada.

Siguiendo el enfoque planteado, se logró implementar un sistema en capas que consolida la información de la oferta de productos de dos supermercados uruguayos: Devoto y Tata. La principal funcionalidad ofrecida consiste en la especificación de un producto por parte de un usuario y en base a esta la determinación de la compra óptima, su lugar, la cantidad de productos disponibles del pedido dado y el precio total, así como por ítem del mismo.

Se cree que este producto aporta valor al público en general y permite, haciendo uso de técnicas de recuperación de información en la web, generar información de utilidad para los usuarios.

Para la implementación de dicho sistema, debieron ser integradas técnicas de diversa naturaleza, presentando cada capa del mismo distintas dificultades y desafíos que debieron ser resueltos. En particular, se encontraron dificultades en cuanto a la calidad de los datos obtenidos, los métodos de recuperación de información debido a la heterogeneidad de las páginas de las fuentes y el procesamiento de los mismos también principalmente por su heterogeneidad y falta de uniformización. Los elementos antes mencionados crearon a su vez dificultades en las capas superiores, por ser los primeros insumos para las mismas. En la siguiente sección detallamos algunos de los problemas encontrados y proponemos líneas de trabajo para mejorar el producto propuesto.

11. Trabajo Futuro

Explicamos en esta sección posibles líneas de trabajo futuro que pretenden generar una mejora en la calidad del producto y la experiencia del usuario en la utilización del mismo.

En cuanto a la calidad de los datos, mediante la incorporación de la fuente Tienda Inglesa se aumentaría la completitud de los mismos, brindando esto una mayor oferta de productos disponibles en los que se realizaría la búsqueda. Esto representa un beneficio al usuario, ya que aumenta potencialmente la oferta de productos, así como la variedad de compras óptimas a ser ofrecidas. El principal desafío encontrado en esta línea es o bien utilizar una herramienta que permita simular el scrolling de las distintas páginas de las categorías de interés para poder cargar todos los productos en la página y luego

ejecutar la spider correspondiente, o bien encontrar una forma de simular los pedidos que genera la página al servidor web correspondiente. Como se reporto en la sección de fuentes de datos, esto presenta dificultades debido a los tokens de seguridad que se intercambian en los pedidos.

Otra mejora posible relativa también a la completitud de los datos considerados, pero en este caso no con respecto a las fuentes si no con relación a la variedad de productos, consiste en incorporar otras categorías que no estén restringidas a productos incluidos en la canasta básica para el consumo. Esto daría mayor utilidad al sistema, si bien se cree que en general la mayor cantidad de productos que son adquiridos por los usuarios cuando estos realizan una compra en un supermercado determinado está constituida por alimentos comestibles y bebidas.

Otro aspecto que redundaría en una mayor usabilidad del sistema consiste en incluir las métricas no normalizadas en el sistema. Muchos productos son adquiridos por los usuarios normalmente en métricas no normalizadas como es el caso de la leche que se adquiere normalmente en litros y no en mililitros. Si bien esto no es un elemento fundamental, ya que la información que se está ofreciendo al usuario es la misma, para algunos usuarios podría no resultar cómodo tener que realizar las conversiones a las métricas normalizadas antes de especificar las cantidades deseadas de un producto que se incluye en el pedido.

Se podrían también incorporar otras variables para la determinación de la compra óptima y no restringirse únicamente al precio total y la disponibilidad de productos para determinarla. Por ejemplo, se podría incorporar la distancia del usuario a los distintos supermercados, que podría ser obtenida por geolocalización en caso de que el dispositivo del que se accede a la página disponga de tecnología GPS. Esto no sería siempre de interés para el usuario, ya que los supermercados también ofrecen la modalidad de envíos a domicilio, mediante la realización vía web de los pedidos. Se ofrecería entonces una funcionalidad que permita especificar por parte del usuario, si es pertinente incluir la distancia al supermercado para definir el mismo. Esto implicaría considerar las distintas sucursales de los supermercados y también idealmente información de stock en cada una de estas, que actualmente no se encuentra en las páginas web de los mismos.

Siguiendo la línea de la usabilidad del sistema, se podrían incorporar las imágenes mostradas en las fuentes en el sistema, con el fin de que el usuario distinga si el producto mostrado por la misma es el que desea o no. Se podrían unificar para los productos que son ofrecidos por más de un supermercado las distintas imágenes utilizadas y mostrárselas al usuario como galería. Esto también ayudaría a identificar si las modalidades en las que son ofrecidos los productos son adecuadas para las necesidades del usuario.

La última área de trabajo futuro que proponemos se refiere a aumentar y refinar las técnicas de procesamiento de lenguaje natural utilizadas en el trabajo. En este sentido, se podría mejorar la uniformización de los nombres de los distintos productos. Esto

permitiría por ejemplo realizar una extracción automática de la marca de los distintos productos, de forma de evitarse la creación manual de diccionarios, que obstaculiza la adaptación automática del sistema frente a incorporaciones, modificaciones o cambios en las marcas ofrecidas por los distintos supermercados. Además, se podrían incorporar métodos que permitan identificar productos que son semánticamente los mismos, aunque posean nombres sintácticamente distintos. Para esto sería necesario generar diccionarios de sinónimos y una vez se genera un pedido por parte del usuario, incluir las variantes pertinentes en la consulta con el fin de identificar todos los productos especificados.

También se podrían incorporar métodos que corrijan consultas que contengan errores para la recuperación de productos. Estas correcciones podrían o bien realizarse de manera automática previo a la realización de la búsqueda, o bien se podría sugerir alternativas al usuario para que el mismo realice la modificación en caso de que lo considere conveniente. Todas estas tareas aumentan en complejidad a medida que la cantidad de productos, así como las fuentes de los mismos se incrementan, por lo que quizás resulte necesario generar un compromiso en la implementación de las distintas mejorar. Podrían entonces implementarse las mismas en etapas con el fin de mejorar de forma progresiva el sistema.

Referencias

[1] Sistema operativo Ubuntu, sitio oficial. Disponible en: https://www.ubuntu.com/, fecha último acceso: Noviembre 2017.

[2] Lenguaje de programación Python, sitio oficial. Disponible en: https://www.python.org/, fecha último acceso: Noviembre 2017.

[3] Framework para desarrollo web Django, sitio oficial. Disponible en: https://www.djangoproject.com/, fecha último acceso: Noviembre 2017.

[4] Framework Scrapy, sitio oficial. Disponible en: https://scrapy.org/, fecha último acceso: Noviembre 2017.

[5] PostgreSQL, sitio oficial. Disponible en: https://www.postgresql.org/, fecha último acceso: Noviembre 2017.

[6] Supermercado Tienda Inlgesa, sitio web. Disponible en: https://www.tiendainglesa.com.uy/, fecha último acceso: Noviembre 2017.

[7] Supermercado Devoto, sitio web. Disponible en: https://www.devoto.com.uy/, fecha último acceso: Noviembre 2017.

[8] Supermercado Tata, sitio web. Disponible en: https://www.tata.com.uy/, fecha último acceso: Noviembre 2017.

[9] Django REST framework, sitio official. Disponible en: http://www.django-rest-framework.org/, fecha último acceso: Noviembre 2017.

[10] R. Fielding, “Architectural Styles and the Design of Network-based Software Architectures”, capítulo 5. University of California, Irvine, 2000.