Recuperación de información y Recomendaciones de la Web

CinematecaFing

Analía Valentina Percovich Hernández	**4.702.590-8**
Bruno Michetti Leites	**4.510.534-4**
Gonzalo Federico Rosso Alfonso	**4.244.387-4**
Alejandro Agustín Tosi Germán	**4.009.267-1**
Timothy Daniel Peraza Geist	**4.539.615-1**

GRUPO 6 - 2017

Índice

Introducción

Hoy en día los sitios web de cines tienen algunas carencias al momento de realizar búsquedas y filtrar preferencias de películas, es por esto que este grupo propone una solución web para unificar los resultados de las tres opciones de cine más grandes que hay en el país:

- Movie - www.movie.com.uy
- Life Cinemas - www.lifecinemas.com.uy
- Grupo Cine - www.grupocine.com.uy

Incluyendo información y filtros que no se tienen en las páginas actuales.

Problema a abordar

Cada cine cuenta con una página web propia en la cual se tiene información de cada uno de sus complejos. A su vez, cada uno de estos complejos exhibe películas en diferentes formatos con diferentes precios.

Es tal, que al momento en que un usuario decide concurrir al cine puede tener problemas para encontrar una función que satisfaga todos sus gustos. Por ejemplo, si se desea saber a qué hora dan cierta película (independientemente del cine) no hay otra forma de saberlo que ingresando a la página web de cada cine recabando dicha información, lo que puede resultar engorroso para cualquier persona.

Por esta razón, la idea que se plantea es que toda esta información se encuentre centralizada en un mismo lugar, de forma que el usuario no deba ingresar a todas las páginas web por separado para encontrar lo que busca.

Solución existente

Actualmente existe una página web que resuelve parcialmente el problema en cuestión:

- Cartelera UY - *cartelera.com.uy*

Si bien esta página centraliza la información de las películas por nombre y por sala, no es posible filtrar, por ejemplo, por género, dimensión o día.

Realizamos un estudio de los filtros e información que posee cada una de las 4 fuentes de funciones de películas y obtuvimos la siguiente comparación:

	Cartelera	Movie	LifeCinemas	GrupoCine
Nombre	Info/Filtro	Info/Filtro	Info/Filtro	Info/Filtro
Días y horarios	Info	Info	Info	Info
Precios		Info	Info	Info
Duración		Info	Info	Info
Género	Info	Info	Info	Info
Ubicación (salas)	Info/Filtro	Info/Filtro	Info/Filtro	Info/Filtro
Snacks y bebidas		Info		
2D/3D/4D	Info	Info/Filtro	Info	Info
Español/ Subtitulada	Info	Info/Filtro	Info	Info
Promociones		Info	Info	Info

A simple vista podemos ver que son muy pocos los filtros que se le proporcionan al usuario, aunque la información de cada una es bastante rica.

Solución propuesta

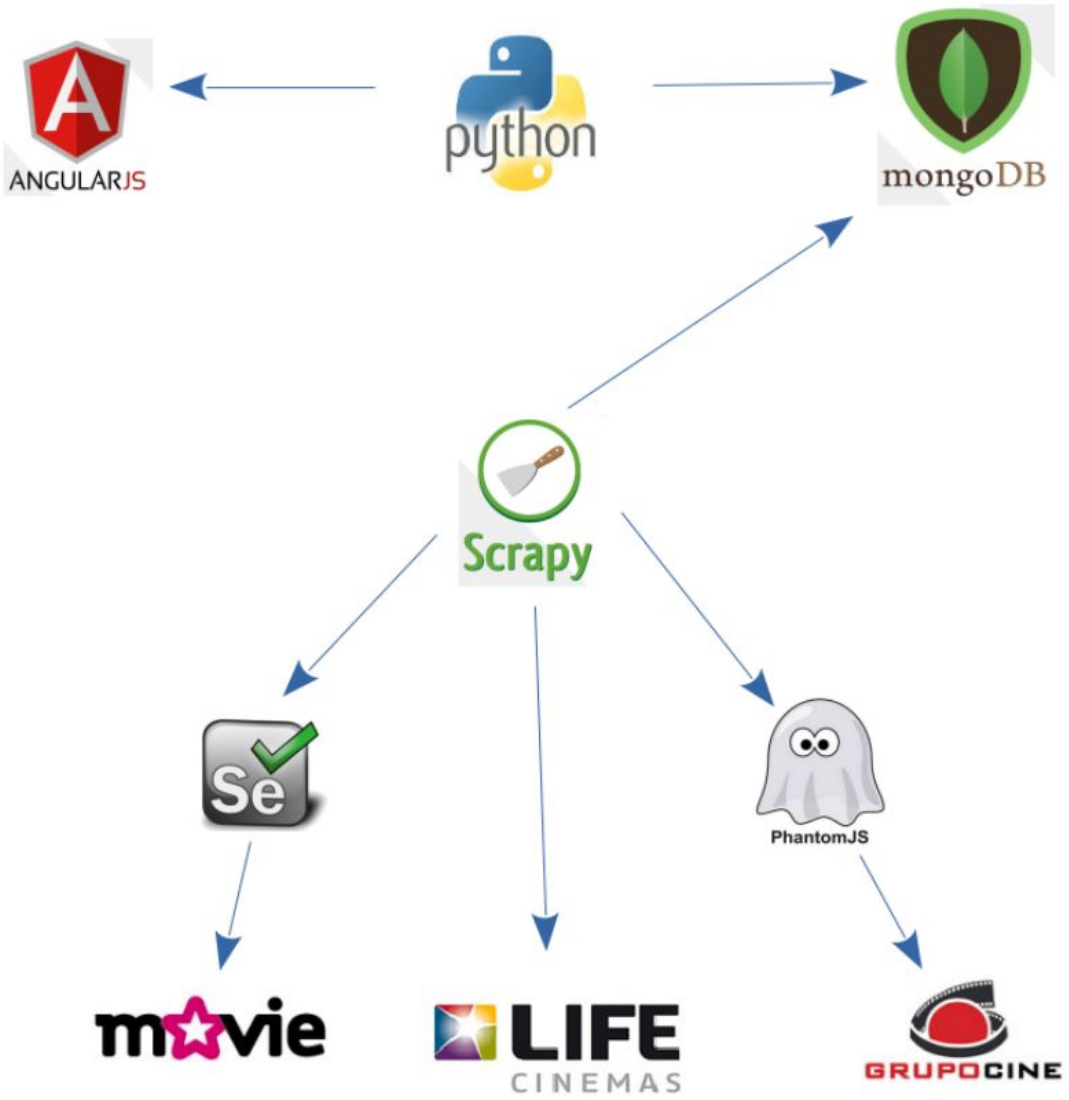
La solución propuesta se llama *CinematecaFIng* y tiene la información de las páginas de las tres fuentes sin tener en cuenta CarteleraUY. Esta nueva solución obtiene la información de todas las películas disponibles en las tres páginas mencionadas, y ofrece un motor de búsqueda que permite filtrar por:

- Nombre de la película.
- Fecha.
- Hora.
- Lugar (cine y complejo).
- Formato (2D, 3D).
- Idioma (Español o subtitulada).
- Género.
- Precio máximo.

Es decir que ingresando/seleccionando los filtros mencionados (todos o algunos de ellos), se retornan todas las películas que cumplen dichos requisitos. Para cada una de ellas se muestra la información disponible (póster de la película, actores principales, duración, etc), así como también las funciones en cada complejo con sus respectivos precios y formatos. A continuación se darán los detalles de la implementación.

Diseño

Arquitectura



Herramientas

AngularJS

Para la implementación del frontend de la aplicación se utiliza AngularJS en su versión 1.6.6. La elección de esta tecnología se debe a su facilidad y rapidez de implementación. Esta aplicación sirve para mostrar los datos de forma amigable y permitir que el usuario pueda hacer uso de los distintos filtros de forma sencilla. Los datos (en formato JSON) a mostrarse se obtienen a partir de API REST provista por el backend realizado en Python.

Python

Durante el transcurso del desarrollo del sistema se utilizó el lenguaje de programación Python para el backend y web scraping. La implementación del backend expone las funcionalidades necesarias para acceder a la información a través de una API RESTful.

Scrapy

Scrapy permite definir de manera muy simple módulos denominados spiders, los cuales se encargaran de recorrer la página acorde a una lógica definida por el usuario y obtener la información deseada de las mismas. En el caso de cada cine, se implementaron dos spiders: una para obtener los datos de las películas y sus funciones, y otra para obtener los precios de los complejos para cada día/formato. Estas mismas spiders son las que persisten directamente a la base de datos los JSON correspondientes.

PhantomJs

PhantomJS es un browser headless con soporte javascript que no renderiza en pantalla, en otras palabras, no tiene una interfaz de usuario, por lo que aumenta la performance de Selenium considerablemente por encima de los browsers convencionales. Fue utilizado para la página de *GrupoCine* ya que la misma se generaba con Genexus y el código fuente no se mostraba en el clásico navegador.

Selenium

Es una herramienta orientada al testing automatizado de páginas web, pero por sus cualidades de navegación automática y obtención de datos resulta ideal para obtener la información de la página del cine *Movie* obteniendo los nodos relevantes mediante XPath, una vez que se ha ejecutado el javascript que genera la página.

MongoDB

MongoDB es un sistema de base de datos NoSQL (not only SQL), orientado a documentos de código abierto. En lugar de guardar los datos en registros, los datos se guardan en documentos, almacenados como BSON, una representación binaria de JSON. Elegimos Mongodb ya que decidimos tener la información plana (esto es, un registro por cada función de cine) y a su vez es cómodo para mostrarla en el frontend con Angularjs. Utilizamos el IDE Robo3t para interactuar directamente con la base.

Implementación

Se obtuvo la información de cada fuente utilizando la técnica de *web scraping*. Los programas que utilizan ésta técnica, generalmente simulan la navegación de un humano utilizando el protocolo HTTP. La herramienta que utilizamos fue Scrapy para Python.

Para algunas de las fuentes fue necesario complementar con algunos plugins soportados por Python, por ejemplo: PhantomJS, ya que el código fuente no era proporcionado directamente por el navegador, por lo tanto Scrapy no era capaz de procesarlo completamente. También se usó Selenium para poder navegar entre los complejos de uno de los cines ya que no se pudo acceder por una URL única.

Luego de recabar la información mediante Scrapy, realizamos un proceso de normalización de los datos de cada uno de los sitios que analizamos. Para lograr la normalización se definieron ciertos criterios de cómo se debía guardar la información, uniformizando todos los datos.

La normalización se realizaba al momento de guardar los datos en los casos que era posible, como por ejemplo pasar el idioma a mayúsculas o cambiar el formato de fechas. Otras tareas de normalización se debieron realizar actualizando la base de datos luego de tener toda la información persistida. Este post-procesamiento se realizó para normalizar datos como los títulos. El proceso consistía en ir recorriendo todos los documentos de la base y para cada película se eligió un título representativo, y utilizando una función de distancia entre strings se definía si un título era lo suficientemente cercano a uno de los representantes como para actualizarle el valor. De esta forma, para cada película y sin depender de qué cine brindó la información, se obtuvo un título único que lograba identificarla. Una vez uniformizada la información en formato JSON proseguimos en guardarla en una base de datos MongoDB.

Finalmente se implementó un front-end basado en una aplicación AngularJS, donde se permite visualizar los resultados obtenidos de las distintas fuentes, pero de una forma amigable. Asimismo, se permite filtrar toda la información por distintos atributos, los cuales creemos pertinentes y que no se hacen presente en las distintas fuentes trabajadas.

La comunicación entre el back-end y front-end se realiza mediante una API implementada en Python.

Problemas encontrados

En esta sección se muestra una síntesis de problemas a los que se enfrentó el grupo de trabajo, que surgieron principalmente de las diferencias entre las formas de cada cine de mostrar la información, y de los diversos lenguajes y aplicaciones de las páginas :

- En algunos casos, para la misma película faltaba información dependiendo del cine, por ejemplo: sinopsis, género, etc.
- Información desnormalizada, por ejemplo: para la misma película, en un cine se encuentra el título en español, pero para otro en inglés y con la traducción en español entre paréntesis. Otro ejemplo es tener información en minúsculas y otra en mayúsculas.

- Distintos formatos de fechas para las funciones.
- La imagen que se mostraba a veces era el póster de presentación de la película, en otros eran fotos de escenas de la película.
- Distintos formatos para la duración, por ejemplo: “115 minutos”, “1h 55m”.
- Palabra abreviadas para presentar el idioma, por ejemplo: “Sub”, “Esp”.
- Los precios en cada cine se muestran, y varían dependiendo de distintos parámetros. Por ejemplo para LifeCinemas dependen del día y complejo, pero en Movie Center dependen del día y formato de la película.
- Los cines GrupoCine y Movie no muestran el código fuente directamente en el buscador, complejizando el proceso de scraping.

Entonces se resolvió que, al momento de persistir los datos de una función de cine, se normalizara la información para simplificar las consultas de los usuarios.

Normalización

En esta sección se enumeran los criterios de normalización de toda la información disponible, de modo de compatibilizar lo obtenido y poder procesarlo. Luego de scrapear las tres páginas, se calculó el número de combinaciones posibles para todas las películas, tomando en cuenta los filtros propuestos y resultó ser aproximadamente 3 mil. Este resultado como tamaño de una base de datos, no es para nada grande, por lo tanto se decidió almacenar en la misma un elemento diferente por cada combinación existente. Esto simplifica extremadamente la búsqueda en la base de datos, y se sabe que no perjudica la performance porque el tamaño máximo posible es 3 mil. En la normalización se pueden visualizar dos partes: pre y post procesamiento.

Pre-procesamiento

Al analizar las fuentes nos encontramos con que las tres páginas actualizan la información todos los miércoles, por lo que decidimos realizar un script con las tres spiders para ejecutar cada vez que llegaba el día de una actualización.

Para el pre-procesamiento de la información, es decir todo lo scrapeado antes de ser guardado en la base de datos, se definieron los siguientes criterios:

- El título de las película se pasa a mayúsculas.
- El nombre de complejo se pasa a mayúsculas, e incluye únicamente el nombre que identifica el complejo. Ejemplos: "*Movie Portone*s" se normaliza a "*PORTONES*" y "*COMPLEJO EJIDO*" a "*EJIDO*".
- El idioma se pasa a mayúsculas y no se usan abreviaciones. Ejemplo: "*Sub*" se normaliza a "*SUBTITULADA*".
- La duración se normaliza al formato "*<cantidad_horas> h <cantidad_minutos> m*". Ejemplo: "*96 minutos*" se normaliza a "*1 h 36 m*".
- La fecha se normaliza al formato "*dd/mm/aaaa*". Ejemplo: "*3/11/2017*" se normaliza a "*03/11/2017*"
- El género se pasa a mayúsculas, y en caso de haber múltiples géneros los mismos se separan por comas. Ejemplo: "*Thriller, Terror*" se normaliza a "*THRILLER, TERROR*", "*ACCIÓN-SUSPENSO-CIENCIA FICCIÓN*" se normaliza a "*ACCIÓN, SUSPENSO, CIENCIA FICCIÓN*"

Post-Procesamiento

El post-procesamiento de la información se hizo actualizando directamente la base de datos. Una de las razones por la que se normalizan ciertos atributos luego de tener toda la información persistida, es para llegar a tener valores coincidentes en los títulos de películas. Para esto se hace necesario contar previamente con toda la información para luego elegir un título base normal a todas las combinaciones de cines, películas, etc.

Se estudió la opción de obtener los títulos de una fuente externa, como de la página Imdb.com, pero surgió que para algunos títulos no se encontraban las películas correctas en dicha página.

Para solucionar este problema se definió la función distancia de Levenshtein, que es llamada por un algoritmo de normalización que recorre la base de datos y según cuán

distantes son los distintos títulos (utilizando un umbral que depende del largo de los títulos), se actualiza el valor de este atributo a uno normalizado.

Luego de una serie de experimentos, se define heurísticamente el umbral que debe no superar una distancia entre títulos para asegurar que se asocian a la misma película, como el máximo del largo entre los 2 títulos divido por 3. Entonces la condición para definir si 2 títulos t_1 y t_2 refieren a la misma película sería:

$$distancia(t_1, t_2) < \frac{max(largo(t_1), largo(t_2))}{3}$$

La otra razón por la que se normaliza en una etapa de post-procesamiento, es debido a la ausencia de información de ciertos atributos para algunos cines, como la sinopsis, los actores, el director y la imagen.

Para esto también se recorre la base y en caso de encontrarse con documentos de funciones obtenidas de cines con atributos faltantes, se busca para el título de la película en cuestión dichos atributos en un documento obtenido de un cine que sí los contenga. Por ejemplo, el atributo sinopsis no se obtiene desde la página de Movie, por lo que es necesario obtener este atributo ya persistido desde la página de Grupocine o LifeCinemas.

Esto presenta el problema de que si una película se exhibe únicamente en un cine de donde no se pueden obtener atributos, entonces la información no se mostrará en las búsquedas por filtros. A pesar de esto, por lo general las películas son encontradas en por lo menos 2 de los cines, lo que hace altamente probable que se puedan conseguir los atributos faltantes.

Estructura del JSON

El JSON que representa cada función en la base de datos está diseñado de la siguiente forma:

```
/* FUNCION DE CINE */
{
      "cine": <valor>,
      "complejo": <valor>,
      "título": <valor>,
      "sinopsis": <valor>,
      "genero": <valor>,
      "calificación": <valor>,
      "duración": <valor>,
      "actores": <valor>,
      "director": <valor>,
      "dia": <valor>,
      "hora": <valor>,
      "dimensión": <valor>,
      "idioma": <valor>,
}
```

En caso de que alguna función no contenga el valor en alguno de sus atributos, éste no aparece en el documento.

Funcionamiento

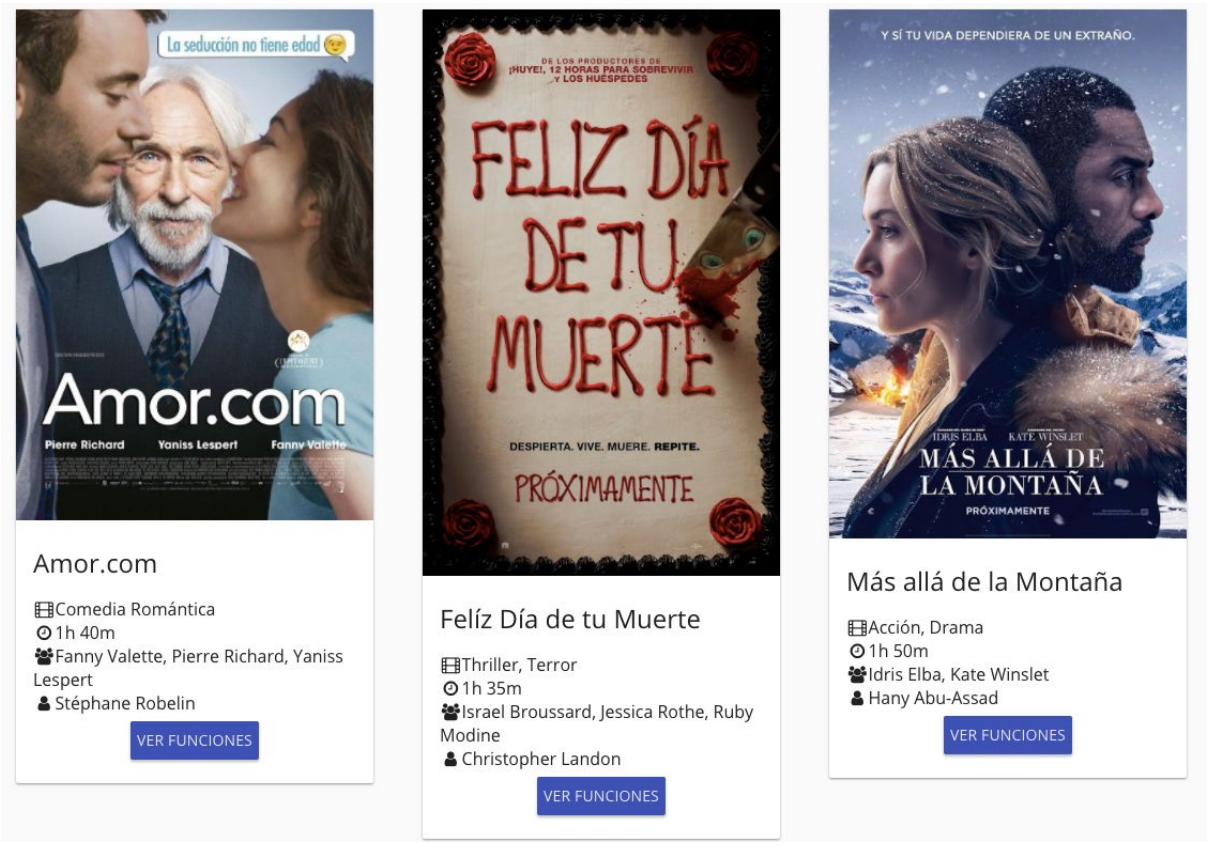
Los usuarios que accedan al sitio web podrán visualizar dos cosas: En primer lugar, una grilla con todo tipo de filtros que el usuario pueda requerir a la hora de buscar una película que desee ver. En segundo lugar una lista de las distintas películas que hay en los tres centros que decidimos analizar (Movie, LifeCinemas y GrupoCine).

Al entrar a la aplicación web, se muestran todas las películas sin aplicar ningún filtro. Una vez el usuario decida realizar una búsqueda podrá ingresar algunos datos en alguno de los filtros proporcionados reduciendo la cantidad de información a mostrar según los datos que complete la persona.

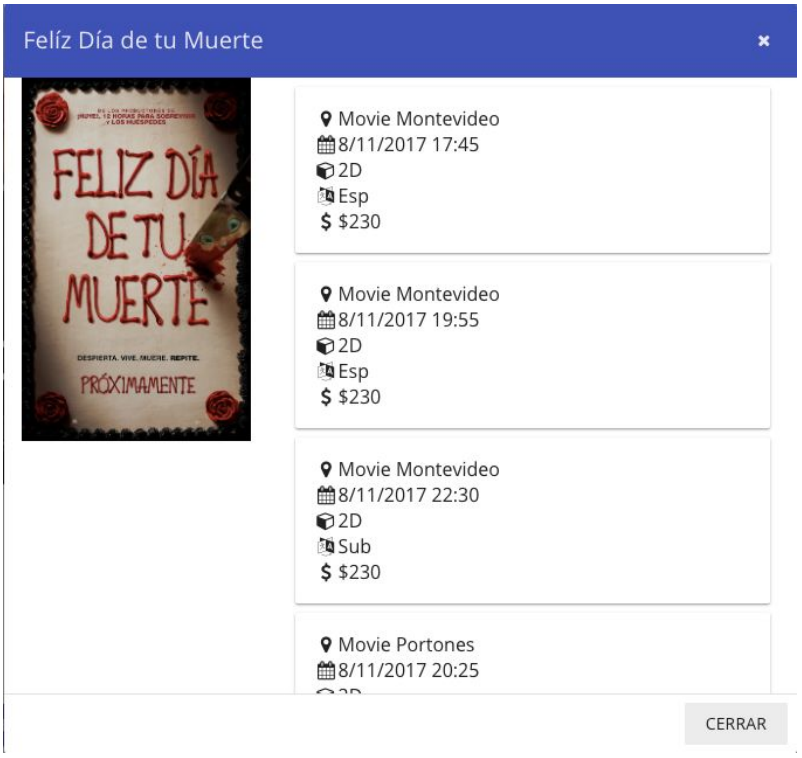
Interfaz

En el diseño de la interfaz se priorizó el destaque de los filtros que se considera que son los faltantes en los distintos sitios de cadenas de cine analizadas.

También se buscó agrupar toda la información referida a cada película bajo una vista común (género, duración, actores, director) y dentro de la misma las distintas funciones con sus detalles correspondientes (complejo, formato 2D/3D, español/subtitulada, precio). Se buscó esta solución para permitir diferenciar rápidamente por película y así evitar confusiones al mostrar varias instancias de la misma película.



En adición a este tema, dentro del detalle de cada función se puede comparar fácilmente el complejo donde se emite y el precio de la función, entre otros detalles relevantes para poder tomar una decisión.



En la siguiente imagen se muestra cómo está diseñada la interfaz completa:

Titulo

Fecha

Precio máximo

Formato

Subtitulada/español

Lugar

Genero

BUSCAR

Amor.com

Comedia Romántica
1h 40m
Fanny Valette, Pierre Richard, Yaniss Lespert
Stéphane Robelin

VER FUNCIONES

Felíz Día de tu Muerte

Thriller, Terror
1h 35m
Israel Broussard, Jessica Rothe, Ruby Modine
Christopher Landon

VER FUNCIONES

Más allá de la Montaña

Acción, Drama
1h 50m
Idris Elba, Kate Winslet
Hany Abu-Assad

VER FUNCIONES

Conclusiones

En esta sección se enumeran conclusiones generales que surgieron al finalizar el proyecto de la presente asignatura. El equipo considera que actualmente las herramientas de obtención de datos de la web son de gran utilidad para la explotación de información que proporcionan distintos sitios de internet pero que tienen ciertos inconvenientes porque (en muchos casos, no en todos) se construyen sin tomar en cuenta la compatibilidad con otras fuentes, y además dichas herramientas dependen mucho de cómo se exponen los datos relevantes. Por ejemplo en el caso de Scrapy, de manera sencilla se pueden obtener los datos expuestos de alguna página pero si la misma en un futuro cambiase su diseño web y la forma en que exponen su información la implementación del spider se volvería prácticamente obsoleta o muy difícil de reajustar.

El transcurso de este trabajo le refleja al equipo la importancia, la utilidad y la dificultad que representa la integración de información de distintas fuentes, ya que cada una expone la información a su gusto y a la hora de querer centralizar hay que realizar muchas decisiones (sobre cómo obtener y normalizar los datos, cómo mostrar los resultados, etc) lo cual no resulta tan simple. Y al igual que en el punto anterior, cualquier cambio de los datos puede afectar el sistema si no se realiza una normalización genérica que pueda representar globalmente a los datos de forma correcta. Se considera que esta área de investigación y de trabajo abre un abanico de diferentes sistemas que se podrían realizar con este tipo de ideas, concentrando en sitios más modernos y legibles para los usuarios diferente información de interés que muchas veces no son fáciles de buscar o están en sitios webs difíciles de comprender ya que: o bien no fueron diseñados correctamente o bien son sistemas legacy u obsoletos para los tiempos modernos.

Finalmente, se puede afirmar que es un trabajo rico en varios aspectos, ya que para llevarlo a cabo fue necesario investigar diferentes herramientas de programación, investigar diversas fuentes de información, investigar sobre la utilidad de la propuesta, tomar decisiones de diseño y enfrentarse a limitaciones que surgen de la integración de datos.

Trabajo a futuro

Se plantean como mejoras a futuro:

- Disponer de una fuente externa con información de las películas, con el fin de poder normalizar todos los atributos, y obtenerlos en caso de que no sea posible conseguirlos en determinadas páginas de cines.
- Agregar nuevos prestadores de servicio de cine como por ejemplo Cinemateca u otros cines de distintas partes de Uruguay, la inclusión de nuevos tipos de espectáculos como por ejemplo Teatro, Carnaval, etc. para poder hacer una sistema mucho más completo para los usuarios finales.
- Mejoras en la web y en el sistema donde se pueda ofrecer mayor información sobre los eventos además de la obtenida desde los proveedores de servicios.
- Desplegar en la nube la aplicación dejándola accesible a cualquier persona.
- Mejorar la búsqueda en la interfaz de usuario utilizando elasticSearch, que provee un motor de búsqueda eficiente de texto completo. En principio el equipo tuvo la intención de incorporarlo, pero luego de investigar sobre la herramienta, se concluyó que era inviable para los plazos de tiempo establecidos.

Referencias

- Python - *www.python.org*
- Scrapy - *scrapy.org*
- Selenium - *www.seleniumhq.org*
- PhantomJS - *phantomjs.org*
- MongoDB - *www.mongodb.com*
- Robo3t - *robo3t.org*