WEBIR - Recuperación de Información y Recomendaciones en la Web

Curso 2020

Trabajo Final

Autores:
Nelson Juambeltz
Javier Perez
Santiago del Pino
Christian Maidana

Profesores:
Libertad Tansini

28 de noviembre de 2020

1. Introducción

En este documento se presenta una solución al problema de una búsqueda de recetas de cocina que incluyan un determinado conjunto de ingredientes, a través de una aplicación web. La aplicación propuesta, ofrece al usuario una búsqueda de recetas media un conjunto de filtros (detallados en la sección 4.Frontend), además de la posibilidad de ver un detalle de cada receta, el cual contiene instrucciones de preparación entre otras cosas.

1.1. Problema

El problema planteado involucra la resolución de diferentes desafíos, tales como, el procesamiento de texto, la obtención y almacenamiento de información disponible en diferentes páginas web, el uso de Elasticsearch para la optimización de consultas, y finalmente la construcción de una API junto con una interfaz web para proveer al usuario de una búsqueda y presentación amigables de receta.

2. Arquitectura de la solución

La solución propuesta cuenta con cuatro componentes principales, los cuales son;

- una base de datos
- un crawler de recetas
- un api para búsqueda de recetas
- una aplicación de frontend

La Figura 1 muestra un diagrama de dicha arquitectura, así como la comunicación entre los módulos de la misma. El crawler de recetas y el api de búsqueda se encapsulan en un mismo módulo por formar parte de la misma aplicación.

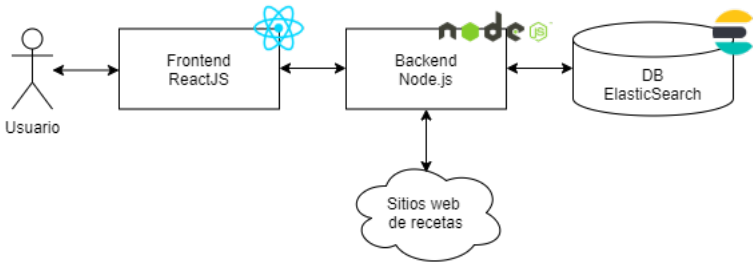

Figura 1: Arquitectura de la solución

3. Backend

El backend es una aplicación programada en NodeJs[1], que consta de dos componentes principales; el crawler de recetas, y el api de búsqueda de recetas.
A continuación, se detallan las consideraciones que llevaron a la elección del lenguaje para el backend. NodeJs es un lenguaje pensado para aplicaciones con gran cantidad de procesos asíncronos, como son los iniciados por el crawler para recolectar información de recetas de la web. Sumado a esto, el procesado de los datos de las recetas previo a su almacenamiento en la base de datos no es lo suficientemente intensivo como para influir negativamente en el desempeño de la aplicación, por lo que no resulta necesario elegir un lenguaje más eficiente, pero de potencialmente más bajo nivel. Por último, el crawler hace gran uso del paquete de node recipe-scraper[2].

3.1. Crawler

El crawler de la aplicación cuenta con dos funciones principales; la recuperación de recetas de la web y su posterior almacenamiento en la base de datos en un formato adecuado, y el pre-procesado de los datos a almacenar, para facilitar la búsqueda.
El pre-procesado de los datos consiste en:

- Convertir el tiempo de preparación del formato “XX hrs YY mins” a un número entero de segundos
- Convertir la cantidad de porciones de string a enteros para admitir una búsqueda por rangos con Elasticsearch
- Remover stopwords de las instrucciones
- Remover palabras que describen a los ingredientes de cocina, pero no son ingredientes

El borrado de stopwords de las instrucciones se hace con la finalidad de evitar resultados incorrectos al matchear stopwords en la búsqueda. El algorítmo utilizado esta basado en la publicación[3], utilizando la lista de stopwords recomendada (la

utilizada por NLTK[4]). Este algorítmo también se utilizo para remover una lista de palabra asociadas a ingredientes, que no constituyen ingredientes en si mismas, como lo son adjetivos y medidas. Esta lista de palabras fue recopilada a mano por el equipo de trabajo, observando recetas ya recuperadas. Dicha lista (no incluyendo las stopwords de NLTK) se adjunta en el apéndice.

3.2. Api de búsqueda

El api de búsqueda genera dinámicamente una query en lenguaje Elastic DSL[5] que fuerza coincidencias en la base de datos para los rangos de porciones y tiempo de preparación total indicados, así como soporte para búsquedas de ingredientes en plural y singular, sin importar cuales el usuario ingrese, a través de filtros wildcard. Finalmente incluimos corrección ortográfica utilizando el soporte de Elasticsearch para búsquedas fuzzy, utilizando una distancia de Levenshtein[6] de 2.

4. Frontend

El frontend es una aplicación programada en ReactJs[7] que provee una interfaz amigable al usuario para interactuar con el api de búsqueda de recetas. ReactJs fue elegida como la tecnología para el frontend, tanto por la familiaridad del equipo de trabajo con la misma, como porque al ser un framework para la construcción de aplicaciones utilizando Javascript, permitió al grupo de trabajo mantener un cierto nivel de coherencia entre el frontend y el backend.
La búsqueda de recetas se presenta al usuario como una barra de búsqueda donde el mismo puede agregar etiquetas de forma aditiva, cada etiqueta correspondiente a un ingrediente distinto que deba contener la receta. Adicionalmente el usuario tiene disponible un filtro para buscar recetas por rangos de tiempo total de preparación y por cantidad de porciones. Las Figura 2 y muestran un ejemplo del funcionamiento de los filtros de búsqueda en la aplicación frontend. También se cuenta con un detalle de cada receta en el que se muestra toda la información disponible sobre la misma, la Figura 3 muestra un ejemplo de esta sección de la aplicación.

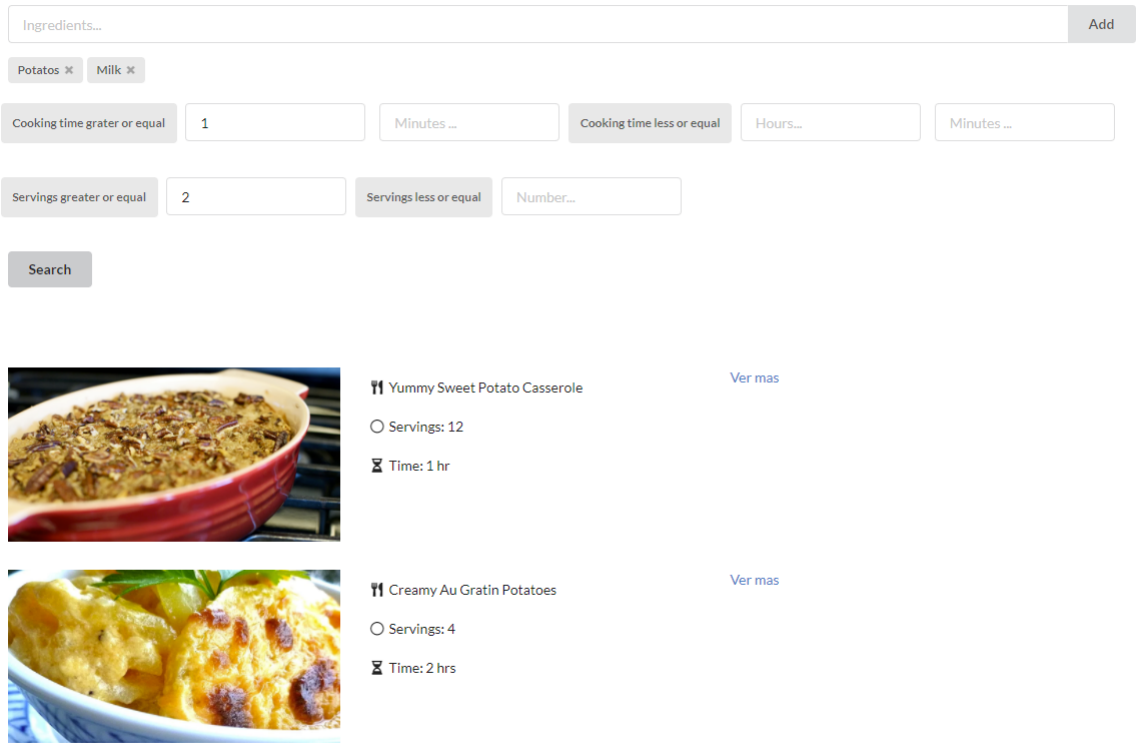

Figura 2: Ejemplo de búsqueda de recetas

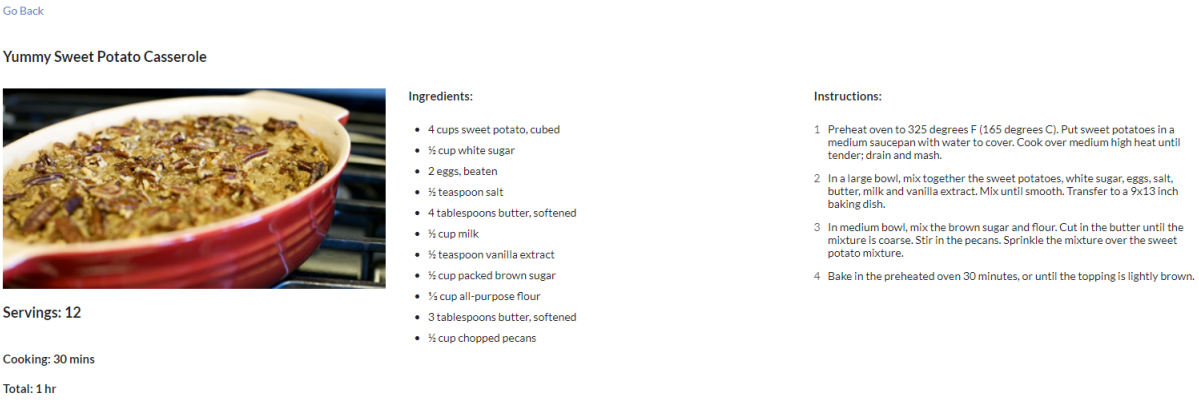

Figura 3: Ejemplo de detalle de receta

5. Base de datos

Para la solución se utilizó una base de datos proveída por Elasticsearch[8]. Los campos relevantes por receta almacenados en la base de datos son:

- Nombre de la receta
- Ingredientes (Lista en lenguaje natural)
- Ingredientes (Lista procesada para facilitar la búsqueda)
- Instrucciones de preparación
- URL de imagen de la receta
- Porciones
- Tiempo total de preparación (En segundos)
- Tiempo total de preparación (En formato XX hrs YY mins)
- Otros tiempos de distintas partes de la receta

La lista de ingredientes se almacena dos veces una en lenguaje natural, para devolver esta información al usuario, y otra vez habiendo sido procesada por el backend, tal como se detalla en la sección 3.Backend, para mejorar la precisión de la búsqueda por ingredientes. A su vez, almacenar el tiempo de preparación tanto en horas y minutos (como lo retorna el crawler) como en segundos, permite hacer búsquedas utilizando los filtros gte y lte[9] para obtener recetas con tiempo de preparación en determinados rangos. Adicionalmente, cada receta tiene un puntaje asociado, calculado por Elasticsearch de relevancia, que ayuda a ordenar las recetas en el frontend de acuerdo a la coincidencia de ingredientes y la relevancia de las recetas.

6. Trabajo futuro

La lista de palabras excluidas de los ingredientes no fue seleccionada a partir de un proceso riguroso. Resulta pertinente el monitoreo de recetas para continuar con la construcción de dicha lista.
La cantidad de filtros disponibles al usuario puede ser expandida para generar una búsqueda aun mas precisa y personalizada.
Finalmente, el grupo de trabajo concentro sus esfuerzos en los aspectos de recuperación de información de la aplicación, por lo que el frontend presenta estilos básicos que pueden ser mejorados, así como la falta de paginación de las recetas.

Referencias

[1] O. Foundation, "Node.js." (28 de noviembre de 2020).

[2] J. Adkins, J. Taranto, L. Robledo, and E. King, "recipe-scraper - npm." (28 de noviembre de 2020).

[3] Cybernetic, "analytics - Stop word removal in Javascript - Stack Overflow." (28 de noviembre de 2020).

[4] N. Project, "Natural Language Toolkit — NLTK 3.5 documentation." (28 de noviembre de 2020).

[5] Elasticsearch, "Query DSL — Elasticsearch Reference [7.10] — Elastic." (28 de noviembre de 2020).

[6] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, 1966.

[7] Facebook, "React – A JavaScript library for building user interfaces." (28 de noviembre de 2020).

[8] Elasticsearch, "Open Source Search: The Creators of Elasticsearch, ELK Stack & Kibana — Elastic." (28 de noviembre de 2020).

[9] Elasticsearch, "Range query — Elasticsearch Reference [7.10] — Elastic." (28 de noviembre de 2020).

7. Apéndice

Lista de palabras removidas de los ingredientes.

cup, cups, lightly, beaten, melted, unsalted , softened, fresh, freshly, ground, heavy, grated, crumbs, pinch, package, thawed, mix, large, chuncks, chunck, needed, baking, quartered, whole, finely, leaves, crushed, minced, mince, can, canned, spice, unbaked, pinches, crumbled, peeled, packed, cored, thinly, uncooked, cooked, clove, cloves, sliced, diced, canned, chopped, aged, baked, boiled, breaded, browned, fried, grilled, parboiled, roast, roasted, sauteed, shredded, cut, pickled, tablespoons, teaspoons, tablespoon, teaspoon, ounce, ounces, pound, pounds, taste, ripe, touch, bunch, big, small, medium, wedges, wedged, slice, powdered, powder, seeded, seed, dry, juice, juiced, dried, seasoning, seasoned