Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Sistema para la obtención de precios de alimentos

System for get prices of foods

Nicolangelo Famiglietti

La Laguna, 5 de julio de 2019

Dr. **Casiano Rodríguez León**, con N.I.F. 42.020.072-S, profesor Catedrático de Universidad del área de Lenguajes y Sistemas informáticos adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

Dra. **Coromoto León Hernández**, con N.I.F. 78.605.216-W, profesora Catedrática de Universidad del área de Lenguajes y Sistemas informáticos adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutora

C E R T I F I C A (N)

Que la presente memoria titulada:

"Sistema para la obtención de precios de alimentos"

ha sido realizada bajo su dirección por D. **Nicolangelo Famiglietti**,

con N.I.E. X-7.079.591-F.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 5 de julio de 2019

Agradecimientos

Agradecer primero a mi familia, Dani, Gonzalo, Iván y Tomás motores incuestionables que hicieron que empezara esta hermosa carrera y no desistieran nunca en llegar hasta el final.

Mis compañeros de clase, ellos también son mi familia porque supieron apoyar cada comienzo de jornada y cada final a lo que horas de estudio se refiere. Serán unos grandes profesionales de la informática.

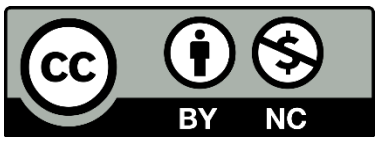
A Almudena, es locura y sensatez que agradezco haber tenido siempre.

A ti Jenni, eres el empujón que me ha hecho falta en los momentos de desánimo y eres la persona que tengo a mi lado en los momentos más importantes durante estos años.

A Casiano y Coromoto por sus conocimientos, calidez humana y gran trato que me han demostraron durante años, desde las asignaturas que me impartieron hasta este Trabajo Fin de Grado. Espero que más alumnos aprovechen las oportunidades que les da el grado con profesores como ellos.

A todos, Gracias.

Licencia



[1] https://creativecommons.org/licenses/by-nc/4.0/

Resumen

En la actualidad existen infinidad de comparadores en línea de servicios diversos, por ejemplo, de seguros de coche, de apartamentos vacacionales o de tarifas telefónicas. También existen de precios de alimentos, pero es más difícil encontrar una API (Application Programming Interface) o servicio web que sea capaz de responder con precios de una determinada categoría de productos, o los precios de un supermercado que opere en Canarias.

Con esta premisa se comienza este trabajo cuyo principal objetivo es obtener precios de alimentos recopilados de las páginas web de los supermercados más habituales que operan en Canarias, concretamente en la isla de Tenerife.

Por ello se ha desarrollado un Sistema Informático, denominado "scra-pi-super" que rastrea el código HTML (HyperText Markup Language) en busca de la información que le interesa y la procesa.

Se ha utilizado Javascript como lenguaje de desarrollo y GitHub como repositorio del código de las distintas versiones.

El sistema scra-pi-super se utiliza desde el sistema CloudCanteen que permite la planificación de menús de comedores escolares.

Palabras clave: Alimentos, salud, planificación de menús

Abstract

There are now countless online comparators for various services, for example, car insurance, holiday apartments or telephone rates. There are also food prices, but it is more difficult to find an API (Application Programming Interface) or web service that can respond with prices of a certain category of products, or the prices of a supermarket that operates in the Canary Islands.

With this premise we start this work whose main objective is to obtain food prices compiled from the web pages of the most common supermarkets that operate in the Canary Islands, specifically on the island of Tenerife.

For this reason, a Computer System has been developed, called "scra-pi-super" that scrape the HTML code (HyperText Markup Language) in search of the information that interests and processes it.

Javascript has been used as a development language and GitHub as a repository of the code for the different versions.

The scra-pi-super system is used from the CloudCanteen system that allows the planning of school lunch menus.

Keywords: Food, health, menu planning

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

La alimentación equilibrada y saludable, complementada adecuadamente con unas pautas de ejercicio físico, sigue siendo la mejor manera de mantener un grado óptimo de salud, lo que se traduce en un peso corporal adecuado según las características de cada persona. Cuando este equilibrio se rompe, sobreviene como primera consecuencia el sobrepeso. Por ello, es importante realizar una elección saludable de los alimentos que componen un menú y tener en cuenta su precio.

Los alimentos, para los seres humanos, son todos aquellos productos que se consumen como fuente de nutrición, de vitaminas, minerales y otros componentes adicionales que les brindan energía y sacian su necesidad de alimentarse. Dentro de este concepto se incluye una variada cantidad de productos que cambian de acuerdo con el tipo de sociedad, la historia, el clima, la tecnología y las costumbres, entre otros aspectos. La importancia de los alimentos radica en que la cantidad y calidad de ellos que una persona consuma, definen en gran medida su calidad de vida y determinan, en cierta forma, su desarrollo físico y cognitivo [5][6].

1.1 Antecedentes y estado actual del tema

Cuando se hace referencia al "Web Scraping" *(raspado web)*, se está haciendo alusión al proceso de extracción y tratamiento de datos, a los cuales normalmente solo se puede acceder al visitar un sitio web con un navegador. Estos datos se obtienen mediante un código que se denomina "raspador". Primero, este envía una consulta "GET" a un sitio web específico. Luego, analiza un documento HTML (HyperText Markup Language)[2] basado en el resultado recibido. Y aquí es donde entra en juego el "Web Scraping" a conveniencia del scraper[3], realizando una extracción de datos de sitios web en un formato utilizable y estructurado. Esto es así ya que cada dato se atasca en una página web, en un formato no estructurado. Al hacer esto de forma autónoma, los scripts de "Web Scraping" abren un mundo de posibilidades como las siguientes:

- Minería y análisis de datos
- Seguimiento de indicadores (inflación, cotización del dólar, Bolsa, etc.).
- Comparación de precios (competitividad).
- Análisis de oferta (competidores, productos y precios).
- Seguimiento de expedientes.
- Evaluación de los consumidores.
- Seguimiento de resultados de búsquedas orgánicas[4].

[2] Lenguaje de marcado para desarrollar páginas web

[3] Persona que diseña un sistema para obtener información de una página web a través de su código HTML

[4] Resultados obtenidos de los diferentes motores de búsqueda sin que influyan agentes externos como la publicidad

Otro término relacionado con "scraping" es "crawler". Un "crawler" busca cualquier tipo de información en la web, por ejemplo: Google, Yahoo, etc., mientras que el "scraping" se centra en determinadas webs y en determinados datos muy específicos. Asociadas a estas técnicas se encuentran las siguientes herramientas:

- GraphQL es un lenguaje de consulta para APIs en tiempo de ejecución y del lado del servidor. No se vincula a ninguna base de datos, sino que se respalda por su código y datos existentes [7].
- REST (Representational State Transfer) para la comunicación entre APIs web utilizando HTTP (Hypertext Transfer Protocol)[5] como protocolo de comunicación [8]
- cheerio.js para analizar y proporcionar una API para leer y manipular la estructura de datos que resulta del proceso de analizar el código HTML solicitado [11].
- pupeteer para realizar la mayoría de las acciones que se pueden realizar en un navegador [9]

1.2 Objetivo

Para reducir el tiempo de búsqueda, poder automatizar la recolección de precios de alimentos y que todo lo haga una misma herramienta, nos fijamos el siguiente objetivo: Crear un servicio web que permita la obtención de precios de alimentos de diferentes páginas web.

1.3 Planificación de tareas

Para cumplir el objetivo y el propósito de este Trabajo de Fin de Grado se organizaron las siguientes tareas:

1. Revisión bibliográfica.
2. Diseño e implementación.
3. Implementación del sistema.
4. Validación y evaluación del sistema
5. Redacción de la memoria y difusión de los resultados

El resto de la memoria se organiza de la siguiente forma: En el capítulo 2 se describen la metodología y herramientas utilizadas. En el capítulo 3 se hace una descripción pormenorizada del sistema. Finalmente, en el capítulo 4 se encuentran la instalación y el modo de uso.

[5] Protocolo de transferencia de hipertexto

Capítulo 2
Metodología y herramientas utilizadas

En este capítulo explicaré las herramientas que se han utilizado en el desarrollo de la aplicación además de las técnicas de trabajo que se han empleado durante estas semanas

2.1 Metodología

La metodología utilizada para la realización de este proyecto fue la metodología ágil de "Extrem programming" *(programación extrema)*.

Las Metodologías ágiles son un enfoque del desarrollo de software centrado en las personas y en los resultados que respeta nuestro mundo que cambia rápidamente. Se centra en la planificación adaptativa, la autoorganización y los cortos plazos de entrega. Es flexible, rápido y apunta a mejoras continuas en la calidad utilizando herramientas de Extreme Programming.

- La metodología ágil, busca implementar el primer incremento en el desarrollo de un software en un par de semanas y toda la pieza de software en un par de meses.
- Los equipos ágiles dentro del negocio trabajan juntos todos los días en cada etapa del proyecto a través de reuniones cara a cara. Esta colaboración y comunicación aseguran que el proceso se mantenga en el camino incluso cuando cambian las condiciones.
- En lugar de esperar hasta la fase de entrega para medir el éxito, los equipos que aprovechan la metodología ágil realizan un seguimiento regular del éxito y la velocidad del proceso de desarrollo. La velocidad se mide después de la entrega de cada incremento.
- Los equipos ágiles y los empleados se autoorganizan. En lugar de seguir un manifiesto de reglas de la gerencia destinadas a producir el resultado deseado, entienden los objetivos y crean su propio camino para alcanzarlos.

Extreme Programming es un ejemplo de cómo esta metodología puede aumentar la satisfacción del cliente. En lugar de entregar todo lo que el cliente podría desear en el futuro, le ofrece lo que necesita ahora, rápidamente. Extreme Programming se centra en lanzamientos frecuentes y ciclos cortos de desarrollo. Utiliza la revisión de código, la programación en parejas[6], las pruebas unitarias y la comunicación frecuente con el cliente [10].

En el tiempo que se tardó en desarrollar el proyecto las reuniones con los tutores eran prácticamente semanales. En ellas poníamos en valor el trabajo realizado la semana anterior, posibles

[6] Este término se refiere donde dos desarrolladores trabajan utilizando una sola máquina. Cada uno tiene un teclado y un ratón. Un programador actúa como el conductor que codifica, mientras que el otro servirá como el observador que revisará el código que se está escribiendo, revisará y revisará de forma ortográfica, mientras que también averiguará el siguiente paso.

problemas y soluciones que iban surgiendo y finalmente los trabajos futuros que debía realizar para la siguiente reunión. Utilizábamos Google Calendar[7] para programar las diferentes citas, además de dejarlo reflejado en el repositorio de GitHub concretamente en el apartado de Projects dónde teníamos un panel de trabajo en el cual íbamos poniendo las tareas según su estado.

Figura 2-1: Diagrama de un Project de GitHub

En cuanto al trabajo personal a la hora de desarrollar la aplicación si tenía alguna duda o problema recurría a la bibliografía [1-2] [14 - 27], Internet o lo comentaba en las reuniones con los tutores los cuales siempre tenían una solución que darme.

2.2 Herramientas utilizadas

- GitHub: Es un sistema de control de versiones de Git. Con esta potente plataforma podíamos alojar las diferentes versiones del proyecto, tener organizadas las tareas según su estado y además el control de los problemas en el apartado Issues. También se aprovechará para tener una versión HTML de esta presente memoria.
- Javascript: La aplicación está desarrollada en este lenguaje de programación ya que nos servía de puente para trabajar con nodeJS. Es un lenguaje orientado a objetos e interpretado que tiene la capacidad de funcionar en un navegador web o como mencionamos anteriormente en un entorno sin navegador como con nodeJS.
- nodeJS: Este entorno de ejecución para Javascript está construido con el motor Javascript V8 desarrollado por Chrome. Funciona en tiempo de ejecución del lado del servidor, multiplataforma, código abierto y que a nosotros nos servirá para ejecutar la aplicación a grandes velocidades gracias a este motor.
- Npm (npm is not an acronym [4]): Es el sistema de gestión de paquetes de

[7] Calendario de Google

nodeJS de los cuales nosotros hemos utilizado:

  - Cheerio: utilizamos este potente paquete para para recorrer el HTML de la página que estemos rastreando los precios. Cheerio nos permitirá buscar determinados selectores como los que se usan para etiquetar el precio de un producto o su nombre. Una vez tenemos esta información, gran parte del trabajo esta completado.
  - Request: Nos sirve para hacer llamadas HTTP y así poder conectarnos a la página del supermercado para servir su código HTML a Cheerio.
  - Commander: Para toda la gestión por línea de comandos
  - FileSystem (FS): Para la gestión de los ficheros ya sean los de configuración de la aplicación o los que sirven de entrada para añadir un nuevo supermercado.
  - SublimeText: Editor de texto

- Mocha: Mocha es un marco de prueba de JavaScript con numerosas funciones que se ejecuta en Node.js y en el navegador, lo que hace que las pruebas asíncronas sean simples [11].
- Chai: Chai es una biblioteca de aseveración TDD[8] (Test-Driven Development) para nodeJS y el navegador que se puede combinar de manera encantadora con cualquier marco de prueba de JavaScript [12].

[8] Desarrollo dirigido por pruebas. Es una práctica de programación que consiste en escribir primero las pruebas (generalmente unitarias), después escribir el código fuente que pase la prueba satisfactoriamente y, por último, refactorizar el código escrito [13].

Capítulo 3
Desarrollo

En este capítulo se describe la arquitectura del sistema y su modo de uso

3.1 Arquitectura del software

La arquitectura se representa según el esquema de la figura 2.4. La aplicación recupera del fichero del supermercado los datos para conectarse a internet y así poder recuperar las categorías y productos. Una vez los tiene y *parseados* en JSON los introduce u actualiza en el mismo fichero.

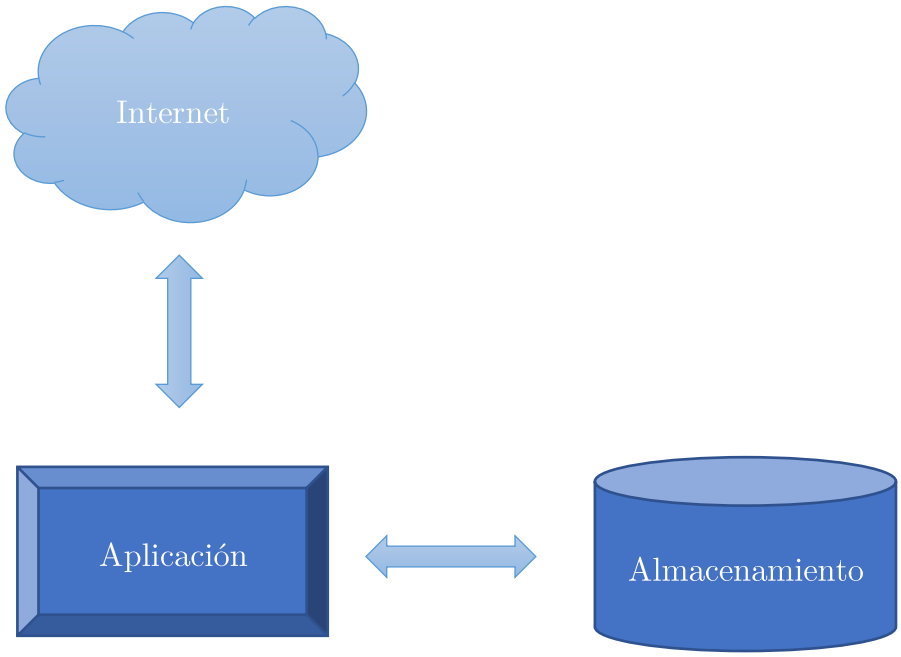


Figura 3-1: Arquitectura del Software

3.1.1 Árbol de directorio

La aplicación una vez instalada crea la siguiente estructura de directorios:

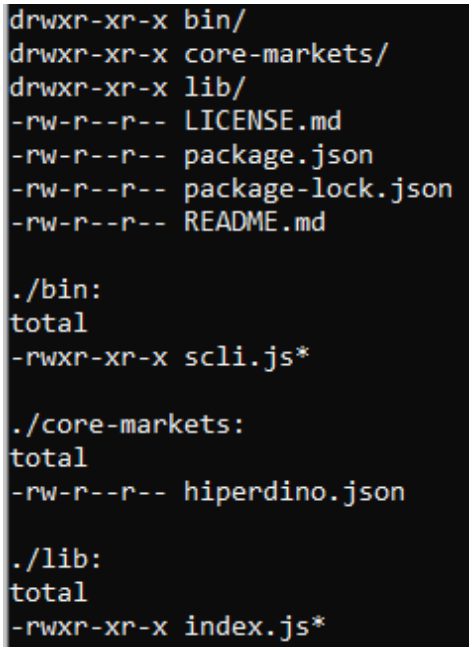


Figura 3-2: Estructura de directorios de la aplicación

- En el directorio *"bin"* contiene los binarios del ejecutable, en nuestro caso la interfaz inicial que contiene las opciones y ayudas de la aplicación.
- En *"core-markets"* alojamos los diferentes ficheros de configuración de los supermercados que contenga la aplicación por defecto siempre que se instale.
- *"lib"* contiene *index.js* con el código fuente de la aplicación.
- El resto de los ficheros son la licencia de la aplicación, *readme* con la información necesaria para el uso del sistema y el *package.json* con los paquetes necesarios para su funcionamiento.

3.2 El sistema Scra-pi-super

La aplicación está concebida para que sirva de información de precios a otras aplicaciones que lo requieran. Así que ya no solo se trabajó en obtener los precios, sino que también se pensó en una manera sencilla para comunicarnos con la aplicación además una forma de almacenar los datos que sea versátil, cómoda y estándar. Esto lo conseguimos con el formato de texto JSON (JavaScript Object Notation)[9] que es muy fácil de tratar, compartir e importar a cualquier base de datos o sistema de datos. Los resultados que se vuelcan al consultar la aplicación vienen en este formato.

Teniendo JSON como formato de respuesta, JavaScript como lenguaje de programación y nodejs como motor de ejecución solo quedaba diseñar las opciones que debían tener la aplicación. La aplicación debía antes que nada poder conectarse al

[9] Notación de objeto de JavaScript

supermercado objetivo y en él poder navegar por su código HTML para rastrear aquellas etiquetas que nos interesaban. Estas son las etiquetas que hacían referencia al precio y al nombre del producto. Para saber el nombre de las clases de estas etiquetas, no quedaba otra opción que mirar manualmente el nombre. Así que estos datos (dirección URL del supermercado y los nombres de las clases de las etiquetas de precio y nombre) son requisitos indispensables para que Cheerio pueda trabajar y darnos esta valiosa información.

Una vez tenemos los productos se *parsean*[10] en JSON y se almacenan en un fichero con el nombre del supermercado con extensión JSON. Junto a los productos, almacenamos la información necesaria para *scrapear*[11] el supermercado. En la figura 2.2 vemos la configuración de Hiperdino y en la 2.3 el resultado de obtener los productos de la categoría "arroz"

Figura 3-3: Configuración de Hiperdino

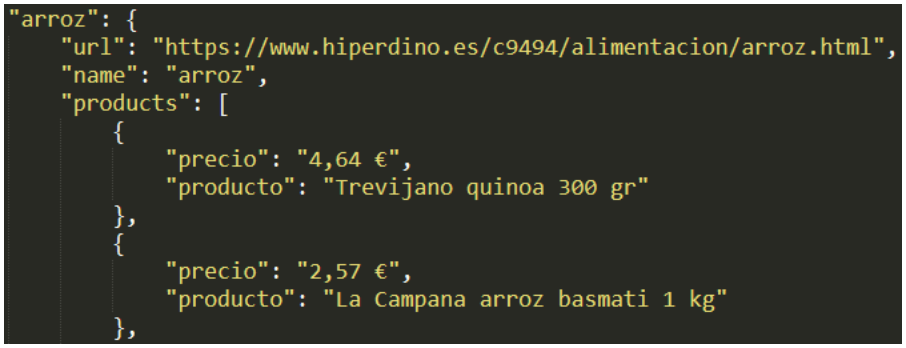


Figura 3-4: Productos de Hiperdino

3.3 Funciones principales

Las principales funciones de la aplicación son:

- addMarket: Añade un nuevo supermercado al sistema. El método de entrada es un JSON

[10] Hacer compatibles datos en un formato con otro diferente.

[11] Término que hace referencia a "rascar" para obtener información de una web

con la información que se refleja en la figura 2.2.

    - -d, --directory: Añade todos los supermercados que encuentre en el directorio.
    - -f, --file: Añade un solo supermercado

- listMarkets: Lista los supermercados que cuenta el sistema para poder recuperar sus precios. Lista únicamente los nombres de estos.
- listCategories: Si se añade el nombre de un supermercado muestra sus categorías, sino lista las categorías almacenadas por cada supermercado configurado en el sistema.
    - Si se le añade el nombre de un supermercado, únicamente visualiza las categorías de este
- listProducts: Añadiendo una expresión regular la aplica a los productos almacenados y muestra los resultados que cumplan esa expresión.
    - -f, --flags: Para añadir flags (en este caso, opciones de adicionales al resultado de la expresión regular)
    - -m, --market: Para especificar en que supermercado buscar, si no se añade esta opción busca en todos los disponibles.
- updatePrices: Se conecta al supermercado, recupera todos los productos que tenga disponible en su web y actualiza los que tenemos almacenados.
    - -a, --all: Actualiza todos los productos de todas las categorías de todos los supermercados
    - -o, --output <file_name>: Almacena el resultado de la actualización en un fichero
    - -s, --screen: Visualiza por pantalla el resultado
    - -m, --market <supermarket>: Almacena los productos de un supermercado
    - -c, --category <category> -m <supermarket>: Almacena todos los productos de una categoría de un supermercado en concreto
- getCategories: Añadiendo el nombre de un supermercado, se conecta a su web y recupera todas sus categorías disponibles en su web.
    - -o, --output <file_name>: Almacena las categorías en un fichero
    - -s, --screen: Visualiza por pantalla el resultado

3.4 Pruebas de validación

Se han desarrollado pruebas unitarias para comprobar sobre todo la disponibilidad de los diferentes portales web en caso de utilizar alguna función del proyecto que necesite conectarse a la página del supermercado y no se obtenga respuesta.

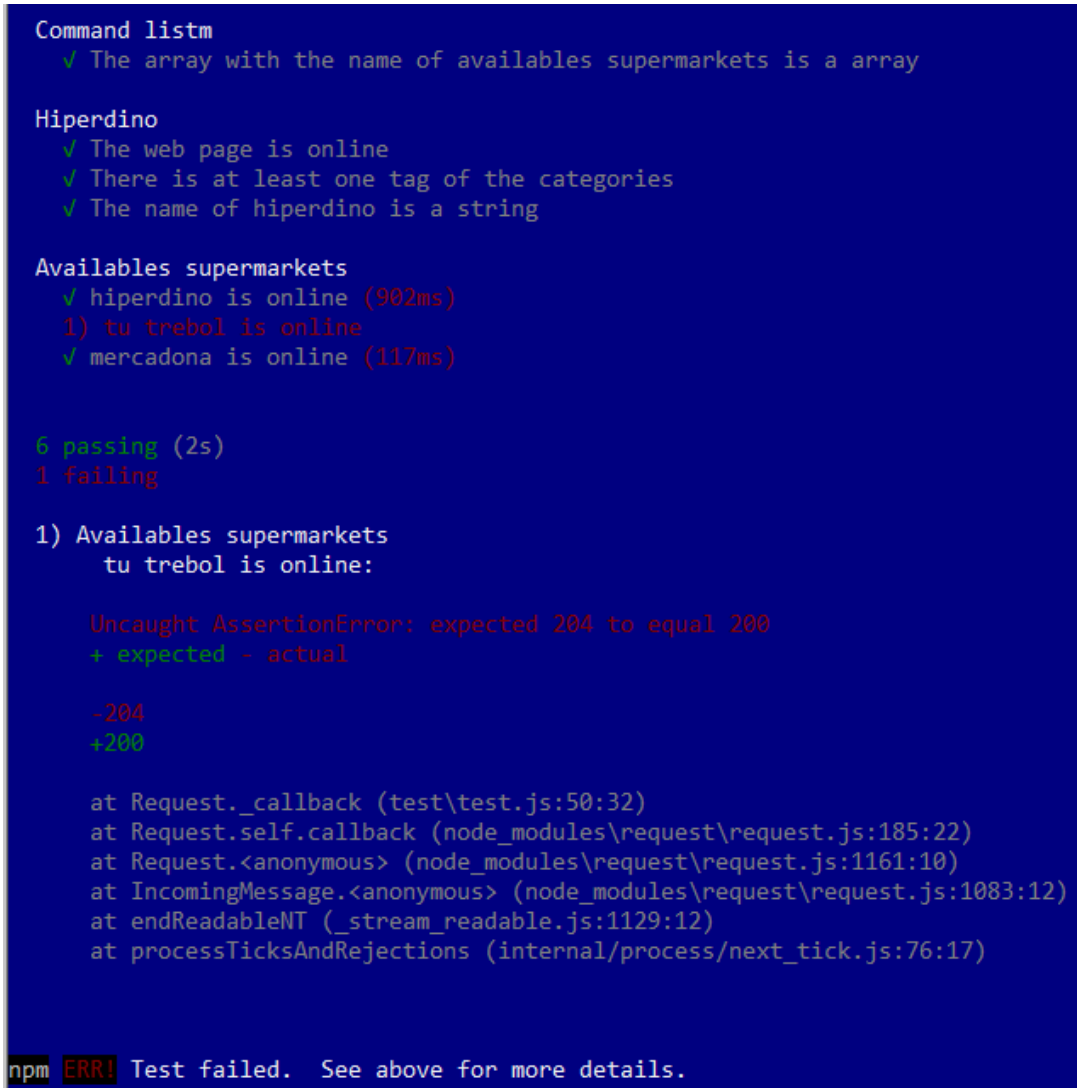


Figura 3-5: Ejecución de las Pruebas unitarias

Capítulo 4

Instalación y modo de uso

En el presente capítulo se explicará la forma de utilizar la aplicación, desde su descarga a su primera ejecución. Además de algunos ejemplos prácticos.

4.1 Instalación

Ya que la aplicación está alojada en npm, la instalamos con su comando de la siguiente forma:

```
$ npm install scra-pi-super -g
```

Tras la instalación y si todo ha ido correctamente al consultar la ayuda de la aplicación nos saldrá la siguiente información:

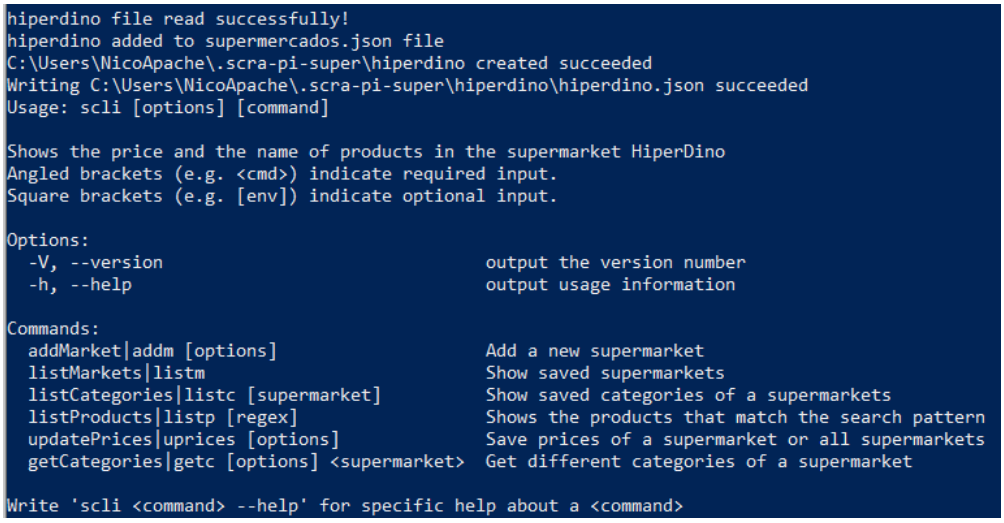


Figura 4-1: Ejecución de la ayuda de la app

Las cuatro primeras líneas podemos ver que ha configurado el supermercado que tiene por defecto en "core-markets". Esto lo realiza automáticamente la primera vez que se ejecuta la aplicación.

4.2 Primeros Pasos

La aplicación tiene por defecto cargada la configuración del supermercado Hiperdino, por lo que es recomendable utilizar el comando que nos busca las categorías de este:

```
$ scli getc hiperdino
```

```
Categories of hiperdino added successfully in C:\Users\NicoApache\.scra-pi-super\hiperdino\hiperdino.json
```

Figura 4-2: Resultado de la ejecución del comando getc hiperdino

Podemos también añadir un supermercado nuevo con el comando:

```
$ scli addMarket [options]
```

Para conocer todas las opciones de este comando, podemos consultar el punto 2.4 de esta memoria.

```
hipertrebol file read successfully!
hipertrebol added to supermercados.json file
C:\Users\NicoApache\.scra-pi-super\hipertrebol created succeeded
Writing C:\Users\NicoApache\.scra-pi-super\hipertrebol\hipertrebol.json succeeded
```

Figura 4-3: Prueba del comando addMarket hipertrebol

4.3 Modo de uso

Una vez tenemos el supermercado cargado con su configuración y además las categorías de este, podemos hacer uso del resto de las opciones.

- ListMarkets

```
pruebas> scli listm
Supermarkets availables:
[ 'hiperdino', 'hipertrebol' ]
```

Figura 4-4: Comando listMarkets

- ListCategories

```
pruebas> scli listc hiperdino
Consulting categories of hiperdino:
aceites
aperitivo
arroz
azucar-y-edulcorantes
cacao-y-cafe
caldos-sopas-y-cremas
cereales
chocolate
comida-internacional
condimentos
conservas-carne
conservas-fruta-y-dulces
```

Figura 4-5: Comando listCategories

- Uprices

```
pruebas> scli uprices -c arroz -m hiperdino
Consulting hiperdino for the category arroz
43 arroz of hiperdino added successfully in C:\Users\NicoApache\.scra-pi-super\hiperdino\hiperdino.json
```

Figura 4-6: Comando uprices

- ListProducts: Para que el comando obtenga resultados debemos tener los productos actualizados (comando uprices) para que haga la búsqueda de la expresión regular entre ellos

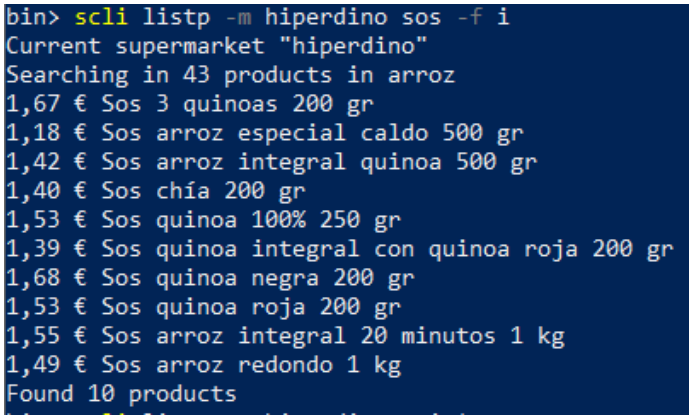


Figura 4-7: Comando listProducts

Capítulo 5
Repositorios y logo

La aplicación está alojada en NPM [3] en el siguiente enlace:

https://www.npmjs.com/package/scra-pi-super

Y en GitHub:

https://github.com/ULL-ESIT-GRADOII-TFG/tfg-nicolangelo-software

Logo de la aplicación:

Figura 5-1: Logo de la aplicación

Capítulo 6
Conclusiones y líneas futuras

Como conclusión principal y gracias a las herramientas utilizadas hemos conseguido de una forma bastante eficiente conseguir lo que se buscaba en el propósito de este Trabajo de Fin de Grado, obtener precios de alimentos. Hemos sido capaces de probar varios métodos de *scrapeo* así como consultar a muchos de los supermercados que trabajan en Tenerife y con más o menos suerte conseguimos desarrollar una herramienta capaz de almacenar en formato JSON los resultados de la búsqueda. Esta búsqueda acepta opciones que nos permite personalizar el rastreo, así como el resultado visualizado o no, almacenado de forma predeterminada o personalizada.

Personalmente si el día de mañana sirve de apoyo a otros sistemas me sentiré orgulloso de los meses que se han invertido en este trabajo. Ser capaces con otras aplicaciones obtener un menú saludable para cualquier persona, pero sobre todo para los más jóvenes y además de conocer de manera actualizada el precio de cada producto que compone dicho menú, hace que esta sea una aplicación que pueda seguir su desarrollo y líneas futuras en otros Trabajos de Fin de Grado.

La línea futura más inmediata sería desarrollar la forma que pueda conectarse de manera automática a los supermercados y así obtener las actualizaciones de los productos o categorías sin necesidad de hacerlo manualmente.

Otra línea futura importante puede ser el desarrollo de *scra-pi-super* como una aplicación REST[12] para que sirva de utilidad a otros sistemas que lo necesiten.

6.1 Summary and Conclusions

As a main conclusion and thanks to the tools used, we have achieved in a very efficient way to achieve what was sought in the purpose of this End of Degree Work, to obtain food prices. We have been able to test several scrapping methods as well as consult many of the supermarkets that work in Tenerife and with luck, we managed to develop a tool capable of storing the search results in JSON format. This search accepts options that allow us to customize the tracking, as well as the displayed result or not, stored by default or customized.

Personally, if tomorrow 'scra-pi-super' will support other systems, I will be proud of the months that have been invested in this work. Being able with other applications to get a healthy menu for anyone, but especially for the youngest and in addition to knowing in an updated manner the price of each product that makes up this menu, makes this an application that can follow their development and lines future in other Final Degree Projects.

The most immediate future line would be to develop the form that can be connected automatically to the supermarkets and thus obtain the updates of the products or categories without having to do it manually.

[12] Capacidad de comunicación entre diferentes aplicaciones mediante el protocolo HTTP

Another important future line may be the development of scra-pi-super as a REST application to be useful to other systems that need it.

Capítulo 7
Presupuesto

En el presente capítulo podremos ver un presupuesto aproximado requerido para la finalización de este Trabajo de Fin de grado

N.º	**Tarea**	**€/horas**	**Horas invertidas**
1	Análisis del campo investigado	11	10
2	Comunicación con los supermercados	11	5
3	Documentación de las herramientas	13	15
4	Reuniones con los tutores	0	20
5	Diseño de la aplicación	15	47
6	Implementación	15	156
7	Pruebas de ejecución	15	32
8	Redacción de esta memoria	13	15
Total		**4080**	**300**

Tabla 1: Tabla de presupuesto

Capítulo 8
Bibliografía

Libros

[1] Flagan, D. (2011). Javascript: the definitive guide. Beijing: O' Reilly.

[2] Bradenbaugh, J. (2000). Aplicaciones en JavaScript. Madrid: Anaya.

Recursos web

[3] Famiglietti, N. (2019, julio). npm: scra-pi-super. Recuperado de https://www.npmjs.com/package/scra-pi-super

[4] Collin, W. (2014, 21 noviembre). npm faq. Recuperado 5 julio, 2019, de https://github.com/google/shipshape/blob/master/third_party/node/lib/node_modules/npm/doc/misc/npm-faq.md

[5] Amador Muñoz, L. V., & Ibáñez, M. (2015, junio). Calidad de vida y formación en hábitos saludables en la alimentación de personas mayores - Artículos - Revista de Humanidades [Investigación]. Recuperado de http://www.revistadehumanidades.com/articulos/95-calidad-de-vida-y-formacion-en-habitos-saludables-en-la-alimentacion-de-personas-mayores

[6] Álvarez Munárriz, L., & Álvarez De Luis, A. (2009, junio). Estilos de vida y alimentación [Investigación]. Recuperado de https://www.ugr.es/%7Epwlac/G25_27Luis_Alvarez-Amaia_Alvarez.html

[7] Facebook. (2015). GraphQL: A query language for APIs. Recuperado de https://graphql.org/

[8] What is REST? | Codecademy. (s.f.). Recuperado de https://www.codecademy.com/articles/what-is-rest

[9] Google Chrome. (s.f.). GoogleChrome/puppeteer. Recuperado de https://github.com/GoogleChrome/puppeteer

[10] Stackify. (2017, 17 septiembre). What is Agile Methodology? Tools, Best Practices & More. Recuperado de https://stackify.com/agile-methodology/

[11] Mocha - the fun, simple, flexible JavaScript test framework. Recuperado de https://mochajs.org/

[12] Chai. (s.f.). Recuperado de https://www.chaijs.com/

[13] Herranz, J. I. (2011, 17 noviembre). TDD como metodología de diseño de software. Recuperado de https://www.paradigmadigital.com/dev/tdd-como-metodologia-de-diseno-

de-software/

[14] Chang, P. (2017, 20 febrero). [Nodejs] Web Scraping note (cheerio). Recuperado 25 febrero, 2019, de https://medium.com/data-scraper-tips-tricks/scraping-data-with-javascript-in-3-minutes-8a7cf8275b31

[15] Desserprit, G. (2017, 27 enero). Scraping data in 3 minutes with Javascript. Recuperado 24 febrero, 2019, de https://medium.com/data-scraper-tips-tricks/scraping-data-with-javascript-in-3-minutes-8a7cf8275b31

[16] Mueller, M. (2017, 1 febrero). cheeriojs. Recuperado 24 febrero, 2019, de https://github.com/cheeriojs/cheerio

[17] Kiessling, M., & Junge, H. A. (2017). El Libro para Principiantes en Node.js» Un tutorial completo de node.js. Recuperado 30 marzo, 2019, de https://www.nodebeginner.org/index-es.html

[18] Rodríguez León, C. (2016). Práctica: Primeros Pasos en NodeJS · ULL-ESIT-1617. Recuperado 30 marzo, 2019, de https://casianorodriguezleon.gitbooks.io/ull-esit-1617/practicas/practicatareasiniciales2.html

[19] Gómez Espejo, N. I. (2017, 2 noviembre). Cómo crear un módulo en NPM. Recuperado 16 abril, 2019, de https://medium.com/@muzk/c%C3%B3mo-crear-un-m%C3%B3dulo-en-npm-11ff8c1c699f

[20] Peraferrer, G. (2015, 22 agosto). Como crear un módulo NPM. Recuperado 16 abril, 2019, de https://medium.com/@peraferrer/como-crear-un-m%C3%B3dulo-npm-6baef161a96

[21] Benjamin, P. (2015, 20 diciembre). Writing Command-Line Applications in NodeJS. Recuperado 20 abril, 2019, de https://www.freecodecamp.org/news/writing-command-line-applications-in-nodejs-2cf8327eee2/

[22] Larrañaga Cahis, F. (2018, 12 junio). Tú, yo y package.json. Recuperado 29 abril, 2019, de https://medium.com/noders/t%C3%BA-yo-y-package-json-9553929fb2e3

[23] Robinson, S. (2017, 2 agosto). Reading and Writing JSON Files with Node.js. Recuperado 30 abril, 2019, de https://stackabuse.com/reading-and-writing-json-files-with-node-js/

[24] Mariuzzo, R. (2017, 18 agosto). A guide to create a NodeJS command-line package. Recuperado 2 mayo, 2019, de https://medium.com/netscape/a-guide-to-create-a-nodejs-command-line-package-c2166ad0452e

[25] Holowaychuk, T. J. (2015). commander.js. Recuperado 13 mayo, 2019, de https://github.com/tj/commander.js/

[26] Cruger, J., Hamann, M., & Robbins, C. (2017, 18 diciembre). npm: nconf. Recuperado 24 mayo, 2019, de https://www.npmjs.com/package/nconf

[27] Riady, Y. (2015, 9 diciembre). A Node.js Configuration Pattern – Yos Riady · Software Craftsman. Recuperado 25 mayo, 2019, de

https://yos.io/2015/12/09/node-configuration-pattern/