Universidad de Oviedo

ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN.

GRADO EN INGENIERÍA EN TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

ÁREA DE TELEMÁTICA

WEB SCRAPING CONFIGURABLE MEDIANTE APLICACIÓN PYTHON

D. GONZÁLEZ GARCÍA, Carmen
TUTOR: D. PÉREZ NÚÑEZ, Pablo
COTUTOR: D. DIEZ PELAEZ, Jorge

FECHA: Febrero, 2023

Resumen

Internet es una de las principales herramientas empleadas para llevar a cabo labores de investigación. En este podemos encontrar numerosas fuentes de información, así como conjuntos de datos predefinidos. Sin embargo, en muchas ocasiones, estos conjuntos contienen más información de la necesaria. En otros casos, el conjunto que buscamos ni siquiera existe.

Ante esta problemática se ha desarrollado la aplicación “MyDataScrape” que, mediante el uso de técnicas de Web Scraping, permite crear conjuntos de datos personalizados por el propio usuario.

Esta personalización se realizará mediante una interfaz gráfica, a través de la cual el usuario podrá especificar, de manera guiada y sencilla, los parámetros para llevar a cabo la búsqueda de información, como la página web o el número de elementos a extraer.

Contenidos

Figuras

Tablas

1. Memoria

Este documento tiene como objetivo explicar los pasos llevados a cabo para el desarrollo de este proyecto, además de la explicación de su funcionamiento y las herramientas empleadas para su creación.

1.1.- INTRODUCCIÓN

La investigación es un proceso que se centra en la búsqueda de nuevos conocimientos y desarrollo de soluciones a los problemas que surgen en todos los ámbitos de la actualidad. Internet es un aliado muy valioso para la investigación, pues ofrece un gran abanico de conjuntos de datos orientados a resolver problemas específicos.

Muchos de estos conjuntos poseen un elevado tamaño, puesto que provienen de la extracción de datos de páginas webs como Wikipedia [1] o Amazon [2].

El tener al alcance conjuntos de datos de un tamaño tan grande, implica, muchas veces, tener que lidiar con una alta cantidad de información irrelevante.

Otro problema es que, a pesar de la variedad, muchas veces no existe un conjunto de datos para el problema que se está intentando resolver. En otros casos, el propio investigador tiene que invertir una gran cantidad de tiempo y esfuerzo en obtener el conjunto que desea.

Con el fin de solventar estos problemas, se ha decidido crear una aplicación que permita configurar y personalizar los conjuntos de datos que un usuario desea obtener de Internet. De esta forma, se podrán crear conjuntos a medida de diversos sitios webs, a los que el usuario tendrá alcance desde su propio equipo.

Así, el usuario podrá llevar a cabo búsquedas de información mucho más efectivas y obtener resultados precisos, sin tener que pasar por cientos de datos irrelevantes.

Esta aplicación será una muestra de las capacidades y ventajas que se pueden llegar a proporcionar, puesto que se permitirá al usuario crear los conjuntos de datos para un limitado número de sitios webs. No obstante, se trata de una aplicación escalable, lo que permitirá ampliar su capacidad y soportar un mayor número de páginas webs en el futuro, necesitando solamente unos simples añadidos en su código base.

Para el desarrollo de esta aplicación se hará uso de la técnica del Web Scraping, cuya traducción al castellano es "Raspado Web". Esta técnica permite extraer datos de sitios web de manera automatizada. Se simula la navegación de un usuario por la web y se recoge la información pertinente.

Para poder acceder a los datos, se hace uso de la estructura HTML de la página web a examinar. La estructura de una página web en HTML está basada en etiquetas, que referencian el tipo de elemento con el que se corresponden, como puede ser un título o una imagen.

Por consiguiente, se utilizarán métodos para localizar cada componente a través de sus referencias en HTML y así obtener la información que está ligada a dicho componente, que es lo que se encuentra visible en la página web.

1.2.- OBJETIVO Y ALCANCE DEL PROYECTO

Los objetivos perseguidos en este proyecto son:

- Desarrollar una aplicación utilizando lenguaje de programación Python para hacer consultas sobre la información de determinados sitios webs.

- Diseño de una interfaz sencilla y fácilmente entendible para el usuario.

- Permitir al usuario guardar los datos recopilados por la aplicación seleccionando el formato en el que serán exportados.

- Facilitar y agilizar al usuario la recopilación y visualización de datos procedentes de una página web a su elección.

- Permitir al usuario una elección concreta de la información que quiere extraer del sitio web.

- Permitir al usuario la ampliación de las capacidades de la aplicación añadiendo pequeñas modificaciones para posibilitar el soporte más páginas webs.

1.3.- RESUMEN DE LAS FUNCIONALIDADES DE LA APLICACIÓN

A continuación, se detallarán las funcionalidades ofrecidas al usuario de la aplicación:

- El usuario podrá seleccionar, de entre un grupo de páginas webs, aquella de la cual quiera extraer la información.

- El usuario podrá seleccionar, de entre un grupo de categorías para cada web, aquella de la cual quiera obtener la información.

- El usuario podrá seleccionar, de entre un grupo de datos, aquellos que quiera obtener sobre cada elemento de la categoría y web que haya seleccionado.

- El usuario podrá seleccionar de cuántos elementos quiere obtener la información.

- Elección del nombre con el que se guardará el archivo.

- Elección del formato en el que se exportarán los datos recopilados (CSV, Json, Excel).

- Elección de la visualización/no visualización del navegador durante el proceso de descarga de los datos.

- El usuario podrá comprobar en todo momento el estado y progreso de la búsqueda mediante una barra de progreso y un log de texto que mostrará los elementos que se van descargando.

- El usuario podrá interrumpir la búsqueda en cualquier momento durante su ejecución y extraer los elementos que se hayan descargado hasta ese momento.

1.4.- DESCRIPCIÓN DE LA APLICACIÓN

La aplicación consta de 2 partes: la interfaz con la que el usuario interactúa y la parte que lleva a cabo la obtención de los datos.

La interfaz permite que el usuario configure y precise su selección respecto a los datos que quiere obtener. La parte funcional hace la búsqueda una vez el usuario ha seleccionado los datos que quiere obtener y los presenta según el formato que se haya decidido.

Esta aplicación está desarrollada en lenguaje Python, por lo que se precisa que, en el ordenador en el que vaya a ser ejecutada, esté instalada una versión compatible de Python. Además, al tratarse de un lenguaje de programación de alto nivel, es necesario un intérprete para que la computadora pueda entender y ejecutar el código escrito en este lenguaje. Asimismo, para poder hacer uso de la aplicación correctamente, es necesario tener ciertas librerías instaladas.

1.5.- DOCUMENTACIÓN QUE ACOMPAÑA AL PROYECTO

En este apartado se detallarán los documentos que acompañan al proyecto:

- **Memoria**: En esta parte se reúne toda la información descriptiva del proyecto y su desarrollo, información acerca de las metodologías y lenguaje de programación empleados y otros aspectos necesarios para la correcta comprensión del proyecto. Incluye la planificación temporal de todas las

actividades y etapas que componen el proyecto, junto con el reparto de roles para su desarrollo.

- **Presupuesto**: Desarrollo del presupuesto estimado para la realización del proyecto.

- **Requisitos del sistema**: Incluye una descripción del sistema actual y los requisitos que deben cumplir, tanto el sistema sobre el que se ejecutará la aplicación como el futuro usuario de esta. También se incluyen los documentos EVS (Estudio de Viabilidad del Sistema) y ASI (Análisis del Sistema de Información).

- **Diseño del Sistema de Información (DSI)**: Documento donde se detalla el diseño del sistema y de la interfaz con la que interactúa el usuario.

- **Protocolo de pruebas**: Documento que recoge las pruebas a las que será sometida la aplicación y las conclusiones a las que se llegarán a partir de ellas.

- **Manual de usuario**: Documento que proporciona la información necesaria para que un usuario nuevo sea capaz de llevar a cabo todos los pasos de instalación previos y requeridos para un correcto aprovechamiento de las prestaciones de la aplicación.

- **Conclusiones**: Documento final que recopila el proceso de desarrollo del proyecto, analizando dificultades y valorando posibles mejor y ampliaciones futuras.

- **Referencias**: Se enumerarán las fuentes de las cuales se obtuvo la información para el desarrollo de este proyecto.

- **Anexos**: Incluye una explicación de las partes más relevantes del código de la aplicación.

1.6.- INFORMACIÓN ADICIONAL DE INTERÉS

En este apartado se describirán términos que el usuario de la aplicación debería conocer para favorecer su comprensión de esta y facilitar futuras ampliaciones que aumenten su capacidad en cuanto al soporte de un mayor número de páginas webs.

1.6.1.- Programación orientada a objetos (POO)

La programación orientada a objetos [3] [4] es un paradigma de programación que se basa en el concepto de "objetos", que son instancias de clases que contienen datos y métodos para manipular esos datos. Los objetos interactúan entre sí mediante mensajes, y cada objeto tiene su propia memoria y comportamiento. La POO se utiliza para crear programas robustos y fáciles de mantener, ya que permite la reutilización de código. Algunos términos que conviene detallar son:

- **Herencia**: Es un mecanismo que permite que una clase herede las propiedades y comportamientos de otra clase, llamada clase base o superclase. La clase que hereda se llama clase derivada o subclase. La herencia permite crear una jerarquía de clases, donde una clase puede heredar de otra y a su vez ser heredada por otra. Además, permite la reutilización de código, ya que una subclase puede acceder a los atributos y métodos de su superclase, y puede también redefinirlos o añadir nuevos. [5]

- **Polimorfismo**: Es un mecanismo que permite tratar objetos de diferentes clases de manera similar, de manera que se podrá crear un código más genérico y reusable. El polimorfismo se logra mediante el uso de puntos de referencia de una clase base o interfaz, lo que permite tratar a objetos de diferentes subclases como si fueran de la clase base. [6]

- **Encapsulamiento de datos:** Es un mecanismo que permite ocultar la implementación detallada de un objeto y exponer solo una interfaz pública para

interactuar con él. Ayuda a proteger la integridad de los datos y a aumentar la seguridad del sistema, ya que los datos solo pueden ser modificados mediante los métodos específicos proporcionados por la clase. [7]

1.6.2.- Interfaz gráfica de usuario (GUI)

Esta interfaz de usuario es el medio que permite a los usuarios interactuar con el sistema informático mediante elementos visuales, como iconos o botones [8]. Es una alternativa más intuitiva y sencilla a la interfaz de línea de comandos.

Debido a su simplicidad, es la forma más popular de interactuar con los sistemas informáticos. Además, favorece la productividad, ya que, al estar basada en los elementos visuales, permite que los usuarios realicen las tareas de forma más rápida y eficiente.

1.6.3.- Web Scraping

El Web Scraping es una técnica que recopila grandes cantidades de datos de páginas webs de forma automatizada. Este método simula lo que sería la navegación realizada por un usuario cualquiera de Internet. [9]

El proceso comienza por la identificación de la URL de la página web de la que se desee obtener la información. A continuación, es necesario el uso de alguna herramienta, como Selenium [10] o Scrapy [11], para hacer una petición a la dirección de la página web y descargar su código HTML. En este proyecto se hará uso de la librería Selenium para lenguaje Python.

Una vez se tiene este código, el programa lo analiza en búsqueda de la información que se quiera obtener, la cual podrá ser almacenada posteriormente en el formato que desee. Para realizar este análisis e identificar los elementos se hacen uso de métodos de localización a través de las etiquetas en el código HTML.

Sin embargo, el uso de esta técnica puede acarrear problemas, puesto que puede violar la política de uso de algún sitio web o incluso captar información personal de los usuarios de una página web. Es por esto que muchas webs toman medidas de seguridad para protegerse y restringir la aplicación de esta técnica, a través de Captchas o seguimientos de IPs [12].

Además, también cabe la posibilidad de que las propias webs se actualicen y haya modificaciones en el código HTML, lo que supondrá un problema a la hora de localizar cada elemento a través su etiqueta, pues habrá que modificar también el código de programación de la aplicación.

1.6.4.- HTML

HTML (Lenguaje de Marcas de Hipertexto, en inglés HyperText Markup Language) es un lenguaje de marcado que se utiliza para la creación de páginas webs. Estructura el contenido de estas a través de unas marcas llamadas etiquetas, que serán a su vez las que permitan diferenciar entre los distintos tipos de elementos existentes. [13]

Cada etiqueta está rodeada por los caracteres “ < ” y “ > ” y la estructura base de las páginas webs está formada por 3 etiquetas principalmente: <html> , que marca el comienzo de la página; <head>, que se corresponde con la cabecera y descripción de la web; <body>, que, como su nombre indica, contiene el cuerpo de la página y todos sus elementos visibles. Después de esta jerarquía existen etiquetas más concretas, como títulos, imágenes o enlaces. [14]

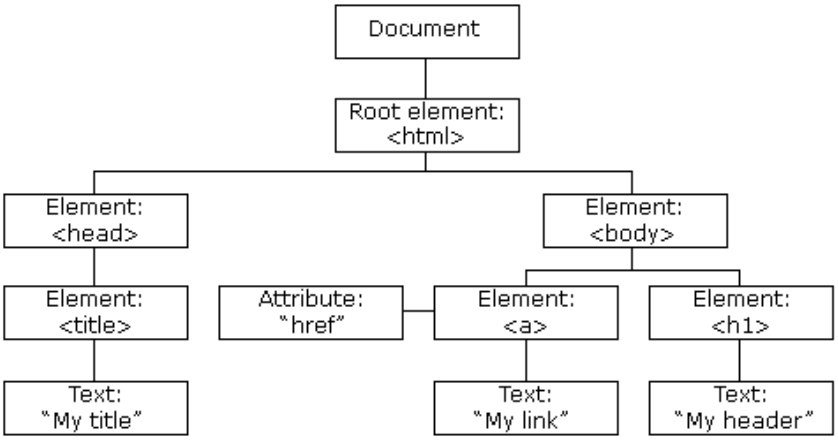

Figura 1.1.- Árbol de objetos HTML. Fuente: w3schools.com [15]

1.7.- DEFINICIÓN DE LAS ETAPAS DEL PROYECTO

El proyecto se divide en las siguientes 6 etapas:

1. Estudio de viabilidad del sistema
2. Análisis del sistema de información
3. Diseño del sistema de información
4. Desarrollo de la aplicación
5. Protocolo de pruebas
6. Documentación del proyecto

1.7.1.- Estudio de viabilidad del sistema

En esta etapa se realizan las tareas iniciales, que están relacionadas con la descripción del proyecto y la evaluación del ámbito y alcance en los que será desarrollado. También se precisarán los usuarios participantes y los requisitos, funcionales y no funcionales, que deberá cumplir la aplicación final.

Para la realización del estudio de viabilidad del sistema se estima un tiempo de **80 horas**.

1.7.2.- Análisis del sistema de información

Se detallará el sistema en su completitud: La estructura final que adquirirá, cada subsistema que conformará el sistema final, los usuarios finales de la aplicación y las interfaces de usuario a través de las que se comunicarán con la aplicación.

Para la realización del análisis del sistema de información se estima un tiempo de **160 horas**.

1.7.3.- Diseño del sistema de información

En esta etapa se establecerá la estructura del sistema y se diseñará en detalle cada parte que lo conforma. También se determinarán las herramientas y tecnologías que se utilizarán para desarrollar la aplicación.

Para la realización del diseño del sistema de información se estima un tiempo de **175 horas**.

1.7.4.- Desarrollo de la aplicación

Se procede a la implementación de la aplicación a partir de las decisiones establecidas en las etapas anteriores. Se desarrollarán, en el lenguaje de programación seleccionado, todas las funcionalidades de la aplicación y la interfaz gráfica de usuario.

Esta será la etapa más larga debido a las tareas que conlleva y a los posibles problemas que se puedan desencadenar a lo largo de su ejecución y que no hayan sido previstos en las etapas de organización previas.

Para la realización del diseño del sistema de información se estima un tiempo de **520 horas**.

1.7.5.- Protocolo de pruebas

Una vez se haya completado el desarrollo de la aplicación, se harán las comprobaciones necesarias de que el sistema funciona según lo esperado y que cumple los requisitos pautados en las etapas anteriores.

Para ello, se elaborará un plan de pruebas que incluirá todas las pruebas necesarias para asegurar que el sistema funciona correctamente. Cuantas más pruebas se realicen, más robusto será el sistema. Además, es conveniente que también se hagan comprobaciones por parte de usuarios diferentes al programador, puesto que podrían detectar fallos o problemas que a este se le hayan pasado por alto.

Para la elaboración del plan de pruebas y resolución de incidencias se estima un tiempo de **120 horas**.

1.7.6.- Documentación del proyecto

En esta etapa se organizará y desarrollará toda la documentación que acompañará al proyecto, que incluirá toda la información recopilada durante todo su proceso de creación.

Para la elaboración de la documentación del proyecto se estima un tiempo de **240 horas**.

1.8.- PLANIFICACIÓN TEMPORAL ESTIMADA

Este proyecto, al tratarse de un trabajo de fin de grado, está destinado a ser completado por una sola persona, quien asumirá todos los roles necesarios para llevarlo a cabo. Por lo tanto, una parte de las tareas han sido ejecutadas de manera secuencial, siendo el final de una tarea el punto de partida para la siguiente.

A continuación se muestra una tabla con los tiempos estimados de realización de cada tarea, junto con el total estimado para el proyecto completo, que deriva de sumar los tiempos individuales de cada tarea por separado.

El tiempo estimado para la realización del proyecto es de **1296 horas**, sin embargo, hay que tener en cuenta que, como se ha mencionado anteriormente, algunas de las tareas han sido llevadas a cabo de manera paralela, como se muestra en la Figura 1.2.

Etapa	Duración
Estudio de viabilidad del sistema	80 horas
Análisis del sistema de información	160 horas
Diseño del sistema de información	176 horas
Desarrollo de la aplicación	520 horas
Protocolo de pruebas	120 horas
Documentación del proyecto	240 horas
TOTAL (estimado)	1296 horas

Tabla 1.1.- Planificación temporal estimada del proyecto

A continuación se muestra el diagrama de Gantt, una herramienta que permite mostrar la cronología de todas las tareas que componen el proyecto. Ha sido desarrollado con el programa Microsoft Project.

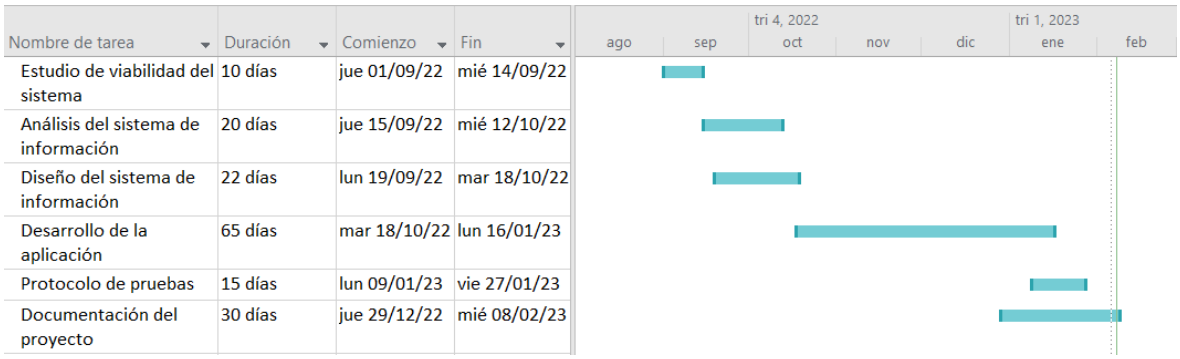

Figura 1.2.- Diagrama de Gantt de la planificación temporal estimada del proyecto

El proyecto se llevará a cabo en un tiempo estimado de **162 días**, con una jornada laboral de 8 horas diarias y teniendo en cuenta que tanto las tareas de Análisis del sistema de información y Diseño del sistema de información, como las de Desarrollo de la aplicación, Protocolo de pruebas y Documentación del proyecto se han realizado, algunos días, de forma paralela.

1.9.- REPARTO DE ROLES

Aunque, como se ha indicado anteriormente, este proyecto haya sido llevado a cabo por una única persona, normalmente un proyecto de este tipo involucra a varias personas, cada una con su correspondiente rol. Estos roles serían los siguientes:

- Director de proyecto.
- Analista.
- Diseñador.
- Programador.
- Usuario "Beta Tester".
- Técnico.

1.10.- DEFINICIÓN DE ROLES

A continuación, se describirán los roles mencionados en el apartado anterior, junto con sus correspondientes funciones y tareas.

1.10.1.- Director de proyecto

El director es el máximo responsable del proyecto y se encarga de que este sea ejecutado de manera eficiente y efectiva. Sus funciones consisten en establecer la planificación y los objetivos del proyecto, fijar los presupuestos, asignar los recursos y supervisar el progreso del mismo.

1.10.2.- Analista

Es el encargado de la realización del Estudio de Viabilidad del Sistema (EVS) y del Análisis del Sistema de Información (ASI).

1.10.3.- Diseñador

El diseñador se encarga de la parte visual del proyecto, la interfaz de usuario. Su tarea será desarrollarla de la forma más acorde a las funcionalidades que la aplicación debe ofrecer, a la vez que mantenga un diseño que facilite su uso y entendimiento a cualquier posible usuario futuro.

1.10.4.- Programador

Es el responsable del desarrollo del código, en el lenguaje de programación requerido, que dotará de funcionalidad a la aplicación. También llevará a cabo la implantación del diseño elaborado por el diseñador.

1.10.5.- Usuario "Beta Tester"

Es el encargado del plan de pruebas que permitirá verificar que el comportamiento y funcionalidades de la aplicación sean las correctas y acordadas, informando de todas las incompatibilidades y fallos que puedan aparecer.

1.10.6.- Técnico

Se encarga del proceso de instalación y puesta en marcha de la aplicación, junto con los manuales de usuario para su correcto entendimiento por parte del usuario final.

1.11.- ASIGNACIÓN DE ROLES

Una vez definidas las tareas a ejecutar y los roles que participarán en su desarrollo, se realizará la asignación entre ambos.

- Estudio de viabilidad del sistema: Analista
- Análisis del sistema de información: Analista
- Diseño del sistema de información: Diseñador, analista.
- Desarrollo de la aplicación: Programador, diseñador.
- Protocolo de pruebas: Usuario "Beta Tester", programador.
- Documentación del proyecto: Técnico.

El director del proyecto se encargará de repartir y supervisar todas las tareas, además de planificar los plazos y presupuestos que se asignarán a cada una de ellas.

2. Presupuesto

2.1.- PRESUPUESTO ESTIMADO

A continuación se muestra una estimación del presupuesto del proyecto completo. Para su desarrollo, se necesita personal, equipamiento hardware y software. Todo el software que se empleará es gratuito, por lo que no se incluirá en el presupuesto.

El presupuesto estimado respecto al personal para el desarrollo del proyecto es el siguiente:

Personal	Días	Horas (8h/día)	Precio/Hora	Precio Total
Director	2	16	30,00 €	480,00 €
Analista	33	264	15,00 €	3.960,00 €
Diseñador	19	152	14,00 €	2.128,00 €
Programador	65	520	14,00 €	7.280,00 €
Beta Tester	13	104	12,00 €	1.248,00 €
Técnico	30	240	12,00 €	2.880,00 €
TOTAL:	**162**	**1296**	-	**17.976,00 €**

Tabla 2.1.- Presupuesto estimado de personal del proyecto

Para llevar a cabo el proyecto se hará uso de un ordenador portátil Acer Aspire A315-54, que tiene un precio de mercado de 609,99 €. Con este equipo no será necesario ningún tipo de material hardware más para completar el proyecto.

Material	Cantidad	Precio/Unidad	Precio Total
Acer Aspire A315-54	1	609,99 €	609,99 €
		TOTAL:	609,99 €

Tabla 2.2.- Presupuesto estimado de material hardware del proyecto

Una vez calculados los presupuestos parciales, se sumarán para obtener el presupuesto total estimado del proyecto. A esta cantidad hay que sumarle un coste añadido correspondiente a gastos generales, que en proyectos de ingeniería suele tener valor del 13%, junto a un 6% de

beneficio industrial. Finalmente, a ese total se le aplica el Impuesto sobre el Valor Añadido (IVA), que asciende al 21%.

Presupuesto Personal …… 17976,00 €
Presupuesto Hardware ……..…… 609,99 €
Presupuesto Software ……..…… 0,00 €
Total costes estimados ……..…… 18585,99 €

Gastos generales (13%) …….. 2416,18 €
Beneficio industrial (6%) ……..…… 1115,16 €

Total sin IVA ……..…… 22117,33 €
IVA (21%) ……..…… 4644,64 €

TOTAL ESTIMADO …… 26761,97

El coste total estimado del proyecto asciende a la suma de veintiséis mil setecientos sesenta y un euros con noventa y siete céntimos.

3. Especificación de requisitos

3.1.- ESTUDIO DE LA SITUACIÓN ACTUAL

Los inicios del web scraping se remontan a los primeros días de la World Wide Web. Con el fin de organizar y rastrear todos los datos disponibles en las primeras páginas webs existentes, se crearon los "web crawlers". Estos web crawlers, o spiders, son programas que navegan automáticamente por la red en busca de información, que posteriormente los principales motores de búsqueda indexarían de forma que pudiesen ofrecérsela al usuario que navegase por Internet. [16]

A medida que Internet crecía y aumentaba considerablemente las cantidades de páginas webs que ofrecían información comenzó a desarrollarse la idea del Web Scraping. Esta técnica sería similar a los ya existentes web crawlers, permitiendo en este caso extraer la información de las páginas webs para así poder almacenarla o analizarla. [17]

Con el tiempo, el Web Scraping se ha ido convirtiendo en una práctica muy habitual, y se han desarrollado una variedad de herramientas y aplicaciones que permitan al usuario aprovecharse de las ventajas que se estas técnicas pueden proporcionar.

Existen múltiples aplicaciones que permitan al usuario extraer datos a partir de una página web. Para ello bastará con que el usuario proporcione la URL del sitio web a analizar y, generalizando el comportamiento típico de las aplicaciones más populares, seleccionar los elementos que se deseen descargar de la página.

Sin embargo, este tipo de aplicaciones no son totalmente accesibles para todo tipo de usuarios, puesto que gran parte de sus funcionalidades suelen ser de pago y, además, debido al mecanismo que utilizan, requieren que el usuario tenga cierto nivel de dominio sobre Internet y la navegación por la web.

3.2.- ESTUDIO DE VIABILIDAD DEL SISTEMA (EVS)

3.2.1.- Ámbito y alcance del proyecto

El objetivo de este proyecto es el desarrollo de una aplicación que permita la creación y exportación de conjuntos de datos personalizables para investigadores. Al realizar una búsqueda en Internet, las cantidades de datos existentes son enormes y pueden contener mucha información que resulte irrelevante.

Por ello, esta aplicación permitirá crear conjuntos a medida para determinados sitios webs que el usuario podrá exportar en el formato que desee. Esta selección podrá ser realizada, de manera simple, por parte del usuario a través de la interfaz gráfica. Además, se tratará de una aplicación escalable, pues se podrá aumentar su capacidad para soportar más páginas webs realizando sencillas modificaciones de su código base.

3.2.2.- Lista de usuarios participantes

Para el desarrollo de este proyecto solamente se contará con 2 tipos de usuarios.

En primer lugar, está el usuario administrador, que será el perfil que forma parte del desarrollo del proyecto y se corresponde con los roles establecidos en el apartado 2.4. Estos serán el director, analista, diseñador, programador, beta tester y técnico. Tendrán permisos para acceder y gestionar todos los componentes de la aplicación, pudiendo hacer modificaciones en caso de necesidad, aunque siempre bajo la supervisión del director del proyecto.

Por otro lado, ya que esta aplicación no requiere ningún tipo de identificación ni registro, únicamente habrá un tipo de usuario del programa. Este podrá hacer uso de todas las funcionalidades de la aplicación, pero sin tener ningún tipo de privilegio más allá de la interacción con la interfaz gráfica, pues no podrá modificar nada en el funcionamiento del sistema ni acceder a su administración.

3.3.- REQUISITOS

Se detallarán los requisitos que debe cumplir la aplicación, que se diferencian en funcionales y no funcionales.

3.3.1.- Requisitos funcionales

Los requisitos funcionales describen el comportamiento esperado del sistema cuando los usuarios interactúan con él. Estos requisitos determinan las funcionalidades y características específicas que se deben cumplir para satisfacer las necesidades de los usuarios.

A continuación se detallan los requisitos funcionales que debe cumplir la aplicación.

RF-1. Ventana principal

Id	Descripción	Prioridad
RF-1.1	El usuario podrá seleccionar la página web de la que obtener datos mediante el desplegable "Página Web.	Alta
RF-1.2	El usuario podrá seleccionar los parámetros propios de la web que haya seleccionado en sus respectivos desplegables. Estos variarán según la web que se haya seleccionado.	Alta
RF-1.3	El usuario podrá indicar el número de elementos de los que quiere obtener información indicándolo en la línea de texto "Número de productos".	Alta
RF-1.4	Si el usuario indica un número de elementos a extraer mayor de los disponibles, se le informará y se descargarán todos los elementos del sitio web posibles.	Alta
RF-1.5	Si el usuario no indica el número de elementos a extraer, o inserta la cifra de 0, se descargará el máximo número de elementos posibles.	Alta
RF-1.6	La interfaz no permitirá introducir caracteres no numéricos en la línea de texto correspondiente al número de elementos.	Baja
RF-1.7	El usuario podrá seleccionar si desea incluir o no ítems patrocinados en la búsqueda.	Baja

RF-1.8	El usuario podrá seleccionar qué datos quiere obtener para la combinación de página web y categoría que haya seleccionado en el cuadro de texto que mostrará todos los disponibles para el conjunto de ambos. En caso de no seleccionar ningún dato, no se podrá hacer la búsqueda y aparecerá un mensaje indicando que se debe seleccionar alguna columna para continuar.	Alta
RF-1.9	El usuario podrá marcar y desmarcar todos los elementos de la lista definida en RF-1.8 automáticamente a través de los botones "Seleccionar todo" y "Deseleccionar todo".	Baja
RF-1.10	El usuario podrá escribir el nombre con el que se guardará el archivo una vez hecha la búsqueda.	Alta
RF-1.11	El nombre del archivo no se podrá repetir entre los ya existentes en el directorio del equipo, por lo que, en tal caso, aparecerá un mensaje informando y el usuario deberá escoger otro nombre.	Baja
RF-1.12	El usuario podrá seleccionar el formato en el que serán exportados los datos mediante el desplegable "Exportar como", teniendo como opción Excel, CSV y Json.	Alta
RF-1.13	El usuario podrá seleccionar la opción de la visualización del navegador mientras se realiza la búsqueda marcando la casilla "Visualizar navegador". En caso de no marcarla, el navegador se mantendrá oculto durante todo el proceso.	Baja
RF-1.14	Al pulsar el botón "Buscar", dará comienzo la búsqueda según los parámetros que el usuario haya seleccionado.	Alta
RF-1.15	Durante la búsqueda, se podrá comprobar su avance mediante la barra de progreso y el cuadro de log en el que aparecerán los elementos que se vayan descargando.	Alta
RF-1.16	Una vez finalizada la búsqueda, aparecerá un mensaje en el cuadro de log indicando que esta ha finalizado, junto al directorio en el que se encontrará el archivo con los datos extraídos.	Alta
RF-1.17	El usuario podrá interrumpir la búsqueda durante su ejecución mediante el botón "Cancelar descarga". Se extraerán los datos descargados hasta ese momento.	Alta

Tabla 3.1.- Tabla de requisitos funcionales de la ventana principal

RF-2. Requisitos generales

Id	Descripción	Prioridad
RF-2.1	El usuario podrá acceder a los datos resultantes de su búsqueda, que se encontrarán en el directorio que se le indicará al finalizar la búsqueda, con el nombre y el formato que haya seleccionado.	Alta
RF-2.2	Los contenidos aparecerán en el archivo exportado ordenados según el identificador de cada elemento de la página web.	Baja
RF-2.3	Se descargará automáticamente el driver "ChromeDriver" en caso de que el usuario no lo tenga instalado en su equipo.	Alta
RF-2.4	Una vez haya terminado la búsqueda, el usuario podrá realizar otra sin necesidad de reiniciar la aplicación.	Baja
RF-2.5	El usuario podrá acceder al archivo de datos cuando haya finalizado la búsqueda, sin necesidad de cerrar la interfaz de la aplicación.	Baja
RF-2.6	La aplicación proporcionará, de base, 3 páginas webs sobre las que realizar las búsquedas.	Baja
RF-2.7	Si el usuario cancela la búsqueda, podrá acceder a los datos que se habían descargado hasta el momento de la interrupción.	Alta

Tabla 3.2.- Tabla de requisitos funcionales generales

3.3.2.- Requisitos no funcionales

Los requisitos no funcionales determinan el comportamiento y diseño que tendrá el sistema, pero sin entrar en especificaciones de las funcionalidades ni las tareas a realizar.

A continuación se detallan los requisitos no funcionales que debe cumplir la aplicación.

RNF-1. Requisitos de usuario

Id	Descripción
RNF-1.1	El usuario debe saber español, puesto que la interfaz gráfica de la aplicación está en este idioma.
RNF-1.2	El usuario debe tener unas nociones básicas sobre informática para saber ejecutar la aplicación.
RNF-1.3	El usuario debe conocer los tipos de formato de archivos Csv,Excel y Json, puesto que tendrá que seleccionar en qué formato desea exportar los datos.
RNF-1.4	El usuario que desee ampliar las capacidades del programa y añadir más páginas webs ha de tener conocimientos de Python, HTML y CSS.

Tabla 3.3.- Tabla de requisitos no funcionales de usuario

RNF-2. Requisitos tecnológicos

Id	Descripción
RNF-2.1	Es necesario un ordenador en el que ejecutar la aplicación.
RNF-2.2	El equipo en el que se ejecute la aplicación debe tener conexión a Internet para llevar a cabo las búsquedas.
RNF-2.3	El equipo debe tener instalada la versión 3.10 de Python.
RNF-2.4	El equipo debe tener instaladas las librerías Selenium, Pandas y PyQt6.
RNF-2.5	El equipo debe tener instalada una versión del navegador Google Chrome.

Tabla 3.4.- Tabla de requisitos no funcionales tecnológicos

RNF-3. Requisitos de usabilidad

Id	Descripción
RNF-3.1	La aplicación debe ser intuitiva y accesible, de modo que los usuarios no necesiten tener conocimientos técnicos para su correcto uso.
RNF-3.2	El proyecto debe contar con un manual de usuario para agilizar su comprensión por parte del usuario.

Tabla 3.5.- Tabla de requisitos no funcionales de usabilidad

RNF-4. Requisitos de escalabilidad

Id	Descripción
RNF-4.1	La aplicación está orientada a la escalabilidad del sistema, necesitando únicamente ligeros añadidos en su código para ampliar su capacidad en cuanto a sitios webs y categorías.
RNF-4.2	El proyecto incluye una guía con los pasos a seguir necesarios para permitir la introducción de nuevas webs junto a sus respectivos parámetros.

Tabla 3.6.- Tabla de requisitos no funcionales de escalabilidad

RNF-5. Requisitos de rendimiento

Id	Descripción
RNF-5.1	La aplicación no debe hacer uso de un excesivo número de recursos del equipo en el que se ejecuta.
RNF-5.2	La aplicación debe ser ligera y proporcionar los datos lo más rápido posible una vez haya finalizado el proceso de búsqueda de los mismos.
RNF-5.3	La interfaz no se bloqueará durante el proceso de descarga de datos, pues este se ejecutará en un hilo secundario.

Tabla 3.7.- Tabla de requisitos no funcionales de rendimiento

3.4.- ANÁLISIS DE ALTERNATIVAS

A continuación se estudiarán las alternativas a las decisiones tomadas a lo largo de este proyecto.

3.4.1.- El mercado de aplicaciones de Web Scraping

En el mercado actual existen varias aplicaciones enfocadas a la aplicación de técnicas de Web Scraping con diversos fines, como investigación, marketing o análisis de tendencias, entre otros.

Se ha realizado un estudio de las principales aplicaciones y se han analizado las más populares en el mercado actualmente.

Respecto a las aplicaciones para scrapear datos de comercios electrónicos, Octoparse [18] es una de las más populares. Su método de funcionamiento consiste en que el usuario introduce la URL del sitio web del que quiere extraer los datos, la aplicación lo carga y comienza a analizarlo en busca de todos sus elementos. Una vez finalizado, ofrece un menú donde se encuentran todos los productos encontrados, con sus correspondientes características, como podrían ser precio o imagen si se tratase de una página web de venta de productos. El usuario podrá extraer los datos que desee de entre todos los que se encuentren. Sin embargo, es una aplicación de pago que ofrece una versión gratuita que limita sus funcionalidades.

Otra opción bastante usada es ParseHub [19]. En esta aplicación, el usuario carga la página web y va seleccionando mediante clicks la información que quiere extraer de ella. A partir de ello, el programa la detecta y la descarga. También es una aplicación de pago, aunque tiene una versión gratuita con funciones restringidas.

La aplicación desarrollada durante este proyecto es una alternativa que, si bien engloba un conjunto mucho más pequeño de opciones, proporciona más facilidades al usuario en cuanto al entendimiento de la aplicación. En las aplicaciones del mercado actual, es el propio usuario el que tiene que aportar la URL e ir marcando manualmente la información que quiera obtener. Esto implica que deba tener unos conocimientos mínimos sobre la navegación por Internet.

Por ello, se creará una interfaz propia, accesible y sencilla, desde la cual pueda comprobar y seleccionar toda la información disponible para extraer, sin necesidad de que tenga que navegar por la página web ni aportar ningún tipo de dato, como la URL.

Simplemente tendrá que rellenar, como si de un cuestionario se tratase, la lista de elementos que contiene la interfaz, marcando las opciones que conformen el conjunto de datos que desee obtener.

Además, los archivos resultantes de la búsqueda se almacenarán directamente en el equipo en el que se ejecute la aplicación, en una carpeta preparada para ello que permita su organización y fácil reconocimiento.

3.4.2.- Lenguaje de programación

Esta es una decisión clave, pues es la base de todo el proyecto y sobre la que se tomarán el resto de las decisiones relevantes para su desarrollo.

Una de las principales características a tener en cuenta es que se trate de un lenguaje orientado a objetos, por las prestaciones que esto nos aportará a la hora de crear la aplicación y sus funcionalidades. Son múltiples los lenguajes que cumplen este requisito: Java, C++, Python y C#, entre otros.

La librería que se empleará para la automatización de la navegación web, Selenium, está disponible para varios de estos lenguajes, Java, Python, C# o JavaScript.

Finalmente, el lenguaje escogido es Python, puesto que es un lenguaje de alto nivel y con una sintaxis clara y sencilla. Además, posee una gran cantidad de librerías y, de entre todos, es el que está más enfocado y tiene desarrolladas más herramientas para el tratamiento y análisis de datos, algo que nos será de utilidad para las funcionalidades de la aplicación y los objetivos del Web Scraping.

3.4.3.- Programa para interfaz gráfica (GUI)

Las opciones que se contemplan a partir de ahora ya estarán condicionadas por el lenguaje de programación previamente escogido, Python. Por ello, para el diseño de la interfaz gráfica se valoran 2 de las principales herramientas compatibles: Tkinter y PyQt.

Tkinter [20] es un paquete estándar de Python para el desarrollo de interfaces gráficas de usuario, ofreciendo una amplia gama de widgets y herramientas para la creación de interfaces personalizadas de manera sencilla.

Por otro lado, PyQt [21] es una biblioteca de código abierto, también para Python, que permite la creación de interfaces gráficas más avanzadas que Tkinter. Ofrece la herramienta QtDesigner [22], que permite incorporar una amplia variedad de elementos, como botones, etiquetas, tablas, listas de datos o campos de texto.

De esta forma, el desarrollador puede ir arrastrando los elementos que desee que compongan la interfaz en una ventana de diseño. Una vez completado, la herramienta generará el código, en lenguaje Python en este caso, correspondiente a dicha interfaz gráfica para su para su posterior aplicación en el proyecto de la aplicación.

Debido a las ventajas que ofrece esta última, será PyQt la herramienta seleccionada para el desarrollo de la interfaz gráfica de usuario.

3.4.4.- Herramienta para la navegación web

Se necesita una librería que permita la automatización de la navegación web para la obtención de los datos. Existen varias opciones para Python, como Playwright [23], que se ejecuta directamente en el navegador y permite automatizar tareas, o Selenium, que permite la creación de scripts para la simulación de las acciones de un usuario en una página web.

La librería Selenium es la más popular y consolidada a día de hoy para la realización de este tipo de tareas, y por ello será la escogida para este proyecto.

Selenium es una librería de código abierta que permite la automatización de pruebas de software para navegadores web. Proporciona una interfaz de programación de aplicaciones (API) que permite a los desarrolladores automatizar la ejecución de pruebas en diferentes navegadores web, como Chrome, Firefox, Safari y Edge.

Selenium hace uso del protocolo WebDriver [24], que se basa en una arquitectura cliente-servidor, donde el cliente (Selenium) envía comandos al servidor (el navegador web), que los ejecuta y devuelve la información solicitada.

El protocolo WebDriver proporciona una interfaz común para interactuar con los navegadores web, estando cada uno respaldado por un controlador propio. El controlador es el componente que se encarga de la comunicación entre el cliente y el servidor, por lo que varía ligeramente según qué navegador se esté empleando. Esta comunicación se lleva a cabo a través del protocolo JSON Wire.

El protocolo JSON Wire [25] es el que se encarga de la transferencia de datos entre el cliente y el servidor. Este protocolo de conexión envía y recibe información a través de peticiones HTTP al controlador. Por tanto, para realizar cualquier interacción con el navegador web, como puede ser localizar un elemento, el cliente (Selenium) envía, a través del protocolo JSON Wire, una solicitud HTTP al controlador web, que será el encargado de realizar la acción que se le ha indicado en la solicitud. Dicha solicitud deberá contener los parámetros necesarios para poder identificar correctamente todos los elementos involucrados. Después, el controlador devolverá el resultado de su tarea mediante una respuesta HTTP.

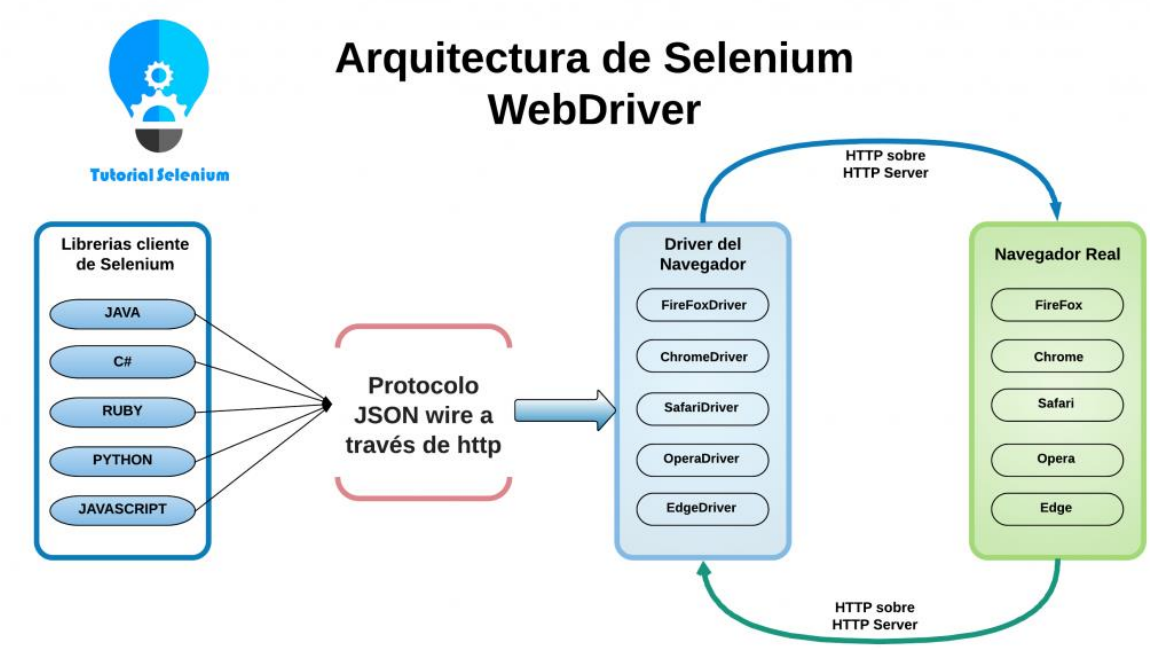

Figura 3.1.- Arquitectura de Selenium WebDriver. Fuente: tutorialselenium.com [26]

El Selenium WebDriver está disponible para múltiples lenguajes de programación: Python, Java, C#, Ruby, Kotlin y JavaScript. Como se ha indicado previamente, este proyecto se desarrollará con el lenguaje Python.

Respecto a los navegadores, el WebDriver soporta: Mozilla Firefox, Google Chrome, Internet Explorer, Microsoft Edge y Safari. Cada uno de ellos tiene su propio controlador, FirefoxDriver, ChromeDriver, InternetExplorerDriver, EdgeDriver y SafariDriver. Para este proyecto se ha empleado el navegador Google Chrome, junto con su correspondiente controlador, ChromeDriver.

Además, existe una librería complementaria, WebDriver Manager, la cual descarga automáticamente la mejor versión del driver correspondiente si el usuario final no lo tiene instalado en su equipo.

3.4.5.- Estructuras de datos

Ya que el objetivo de esta aplicación es la creación de conjuntos de datos, debe escogerse una herramienta que permita su tratamiento y manipulación de una forma sencilla. Existen varias librerías para Python enfocadas al análisis de datos, como Dask [27] o NumPy [28].

Sin embargo, la herramienta por excelencia para Python relativa al trabajo con datos es la librería Pandas [29], y por ello será la que se utilice para el desarrollo de la aplicación.

Pandas es una librería de código abierto de Python que ofrece herramientas para facilitar el análisis y tratamiento de datos. Existen 2 estructuras de datos: Series y DataFrames. [30]

Las series son estructuras similares a arrays unidimensionales que contienen datos de cualquier tipo y cuyos valores se identifican mediante un índice. Los DataFrames son estructuras bidimensionales compuestas por columnas que, individualmente, serían como una serie.

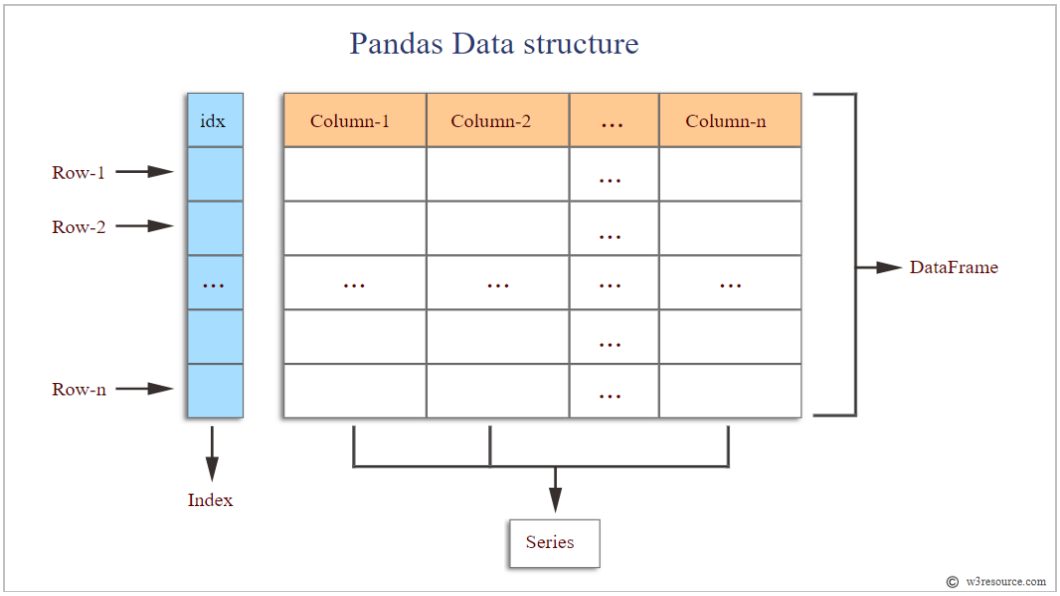

Figura 3.2.- Estructura de datos en Pandas. Fuente: w3resource.com [31]

Respecto a las funcionalidades que nos proporciona esta librería, permite hacer todo tipo de manipulación de los datos, como cargar y exportarlos en distintos formatos, filtrar y seleccionar filas y columnas concretas, combinar DataFrames o realizar operaciones de análisis estadísticos, entre otras muchas cosas.

3.4.6.- Páginas webs para la búsqueda

Esta aplicación tiene como objetivo la obtención de conjuntos de datos procedentes de varias páginas webs. Sin embargo, no todos los sitios webs admiten la realización de las técnicas de Web Scraping, pues algunos toman medidas de seguridad para protegerse contra estas.

Durante la realización de este proyecto se han comprobado la problemática de esto, puesto que se han encontrado páginas que, si bien inicialmente podían ser de uso para la aplicación, finalmente han tenido que desecharse por su imposibilidad de análisis.

Entre estas, se encuentra la página web de Fnac [32], que detecta un comportamiento sospechoso por parte del navegador e introduce una comprobación para detectar que el usuario no se trata de un robot, lo que imposibilita la automatización de la navegación por esta web.

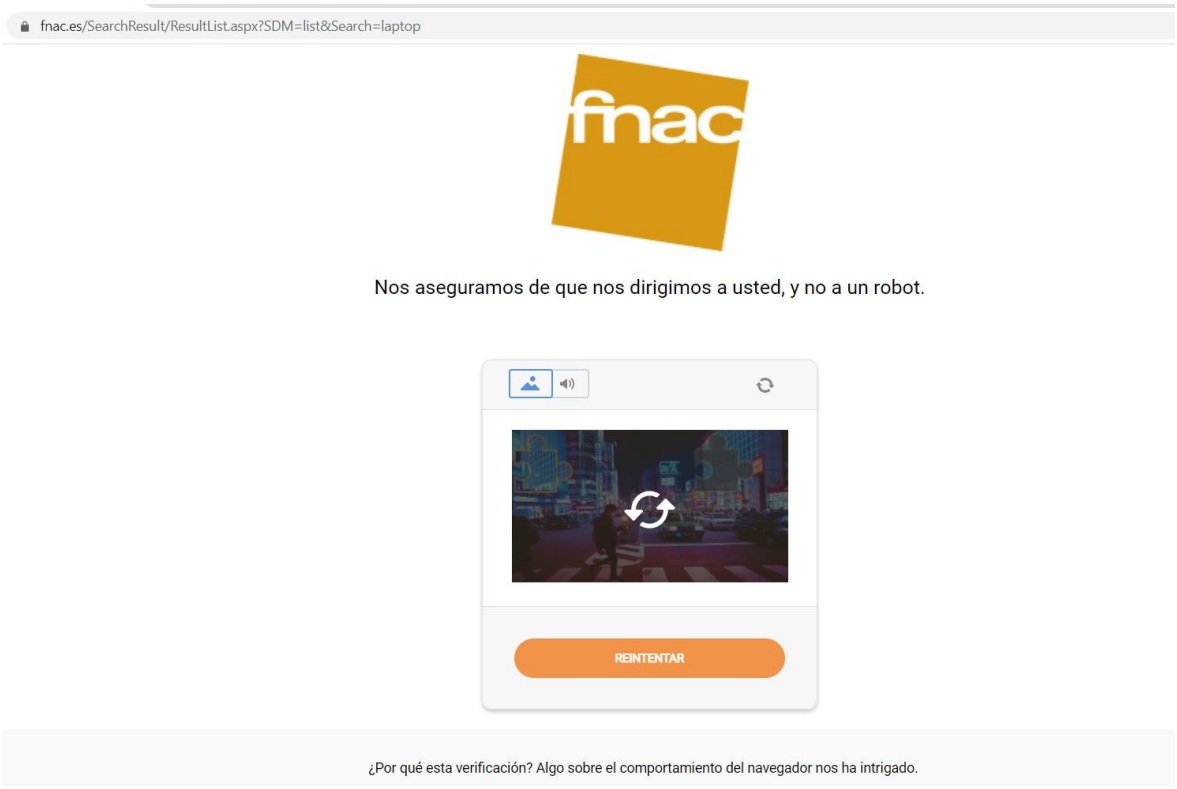

Figura 3.3.- Captura de la verificación de usuario de la página web Fnac.es

Otro problema similar sucede con la página web de Milanuncios [33], que detecta que la navegación no está siendo hecha por un usuario e interrumpe los permisos de actividad por su web.

También hay que tener en cuenta, puesto que es otro inconveniente que se ha encontrado durante el desarrollo del proyecto, que las webs se modifican periódicamente. Esto implica que los selectores que se utilizan para localizar cada elemento cambian, por lo que es necesario actualizarlos en el código de la aplicación para poder mantener el funcionamiento correcto y sincronizado con la versión más actual del sitio web.

Finalmente, las páginas webs que se han seleccionado son: Amazon [2], Tripadvisor [34] y Mediamarkt [35], puesto que se ha comprobado que, al menos durante el proceso de desarrollo del proyecto, soportan la aplicación de las técnicas de navegación automatizada.

Dentro de cada uno de estos sitios webs, se explorarán varias de sus categorías de productos existentes.

3.5.- ANÁLISIS DEL SISTEMA DE INFORMACIÓN (ASI)

Según se indicó en el Estudio de Viabilidad del Sistema (EVS), el objetivo de este proyecto es el desarrollo de una aplicación para la creación y exportación de conjuntos de datos.

A continuación se llevará a cabo un análisis de la aplicación, que permita su posterior diseño de acuerdo con las especificaciones y objetivos fijados.

El diseño de la aplicación de dividirá en 6 subsistemas:

- **Subsistema "Selección de web":** En este subsistema se indica el componente principal del programa: la página web en la que se realizará la búsqueda

- **Subsistema "Parámetros de la web":** Según qué página web se haya escogido, se tendrán en cuenta unos parámetros u otros. Todas las páginas tienen la opción de seleccionar qué categoría de productos es la que se va a extraer. También hay una web para la que existe el parámetro en el que se indique la ciudad en la que se hará la búsqueda. Adicionalmente se podrá indicar si se desea incluir, o no, los elementos patrocinados en la búsqueda de las páginas webs que así lo permitan.

- **Subsistema "Selección de elementos":** Una vez se ha indicado la combinación de página web junto a los parámetros de sus productos, aparecerán listadas todas las características que estén disponibles para descargar, que variarán para cada par página web-categoría. El usuario podrá seleccionar los que desee de entre esta lista y, en caso de que no marque ninguno, aparecerá un aviso indicando que no podrá continuar hasta que seleccione, al menos, 1 elemento. Habrá un botón que permita seleccionar automáticamente todos los elementos, y otro para deseleccionar todo. También deberá indicar el número de elementos que desee extraer, no pudiendo continuar con la búsqueda si no indica esto. Además, si se

pide un número de productos mayor al disponible en la web se indicará y se descargará el máximo número posible de elementos.

- **Subsistema "Exportar archivo":** En este apartado, el usuario podrá indicar el nombre con el que se exportará el archivo tras la búsqueda, junto al formato que adoptará. El nombre no debe repetirse de entre los ya existentes en el directorio, pues se hará una comprobación y se indicará en caso de que ya exista, obligando al usuario a escoger otro que no esté en uso.

- **Subsistema "Búsqueda":** Aquí se encuentra todo lo relativo a la búsqueda de la información. El usuario podrá seleccionar si desea o no visualizar la ventana de navegación. También es donde se encuentra el botón que marca el comienzo de la búsqueda y el que permite cancelarla una vez está ejecutándose. Además, contiene una barra de progreso y un cuadro de texto que se van actualizando a medida que avanza la búsqueda a modo informativo para el usuario.

- **Subsistema "Ventana de navegación":** Es la ventana que muestra la navegación por el sitio web del que se extraerá la información. El usuario podrá escoger si quiere hacer esta ventana visible o no. Sin embargo, esta ventana está controlada por el controlador y, en caso de mostrarla, el usuario no podrá interaccionar con ella en ningún momento.

3.5.1.- Casos de uso

Los casos de uso definen, a partir de los requisitos del programa, las interacciones de los usuarios con la aplicación. A continuación se mostrarán los diferentes casos de uso para cada uno de los subsistemas presentados anteriormente.

3.5.1.1- Casos de uso general

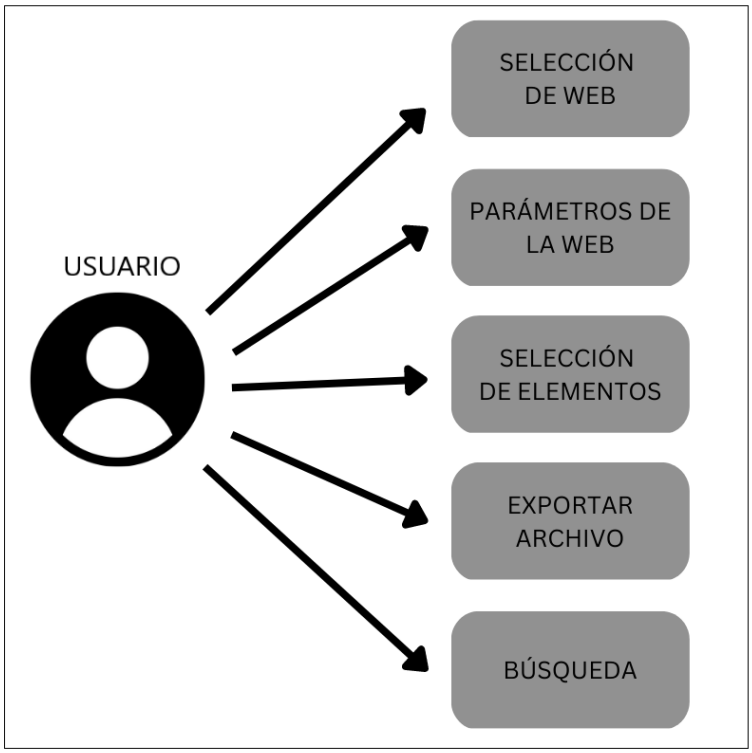

Figura 3.4.- Casos de uso generales

Estos son los 5 subsistemas de la aplicación con los que el usuario puede interaccionar. Cada uno de ellos engloba un conjunto de acciones relativas a su funcionalidad. En los siguientes apartados se detallarán los casos de uso de cada función perteneciente a estos subsistemas.

3.5.1.2- Casos de uso subsistema Selección de web

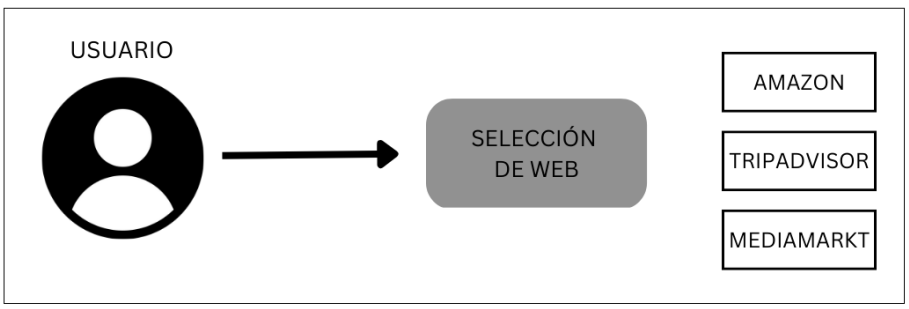

Figura 3.5.- Casos de uso Selección de web

Identificador	CU-1
Nombre	Selección página web
Actores	Usuario
Requisitos	RF-1.1, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Seleccionar la página web de la que se extraerán los datos, de entre las opciones posibles en la aplicación.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas.
Flujo principal	1.- El usuario inicia la aplicación 2.- El usuario escoge la página web mediante un desplegable que muestra todas las opciones disponibles. La opción inicial que aparece por defecto es Tripadvisor.
Excepciones	Las páginas webs que se pueden escoger son 3: Amazon, Tripadvisor y Mediamarkt, pues son para las que está preparada la aplicación. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.8.- Caso de uso selección página web

3.5.1.3- Casos de uso subsistema Parámetros de la web

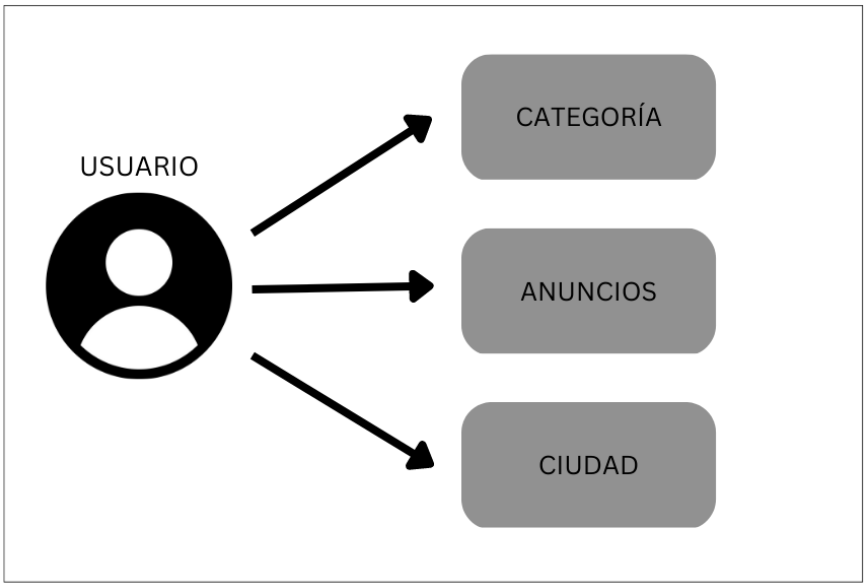

Figura 3.6.- Casos de uso Parámetros de la web

Identificador	CU-2
Nombre	Selección categoría
Actores	Usuario
Requisitos	RF-1.2, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Seleccionar la categoría de la página web decidida sobre la que se extraerá el conjunto de datos.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web deseada.
Flujo principal	1.- Una vez el usuario ha seleccionado la página web que desea, el apartado "Parámetros de la web" se actualizará mostrando todos los parámetros disponibles para esa web. Todas las webs contienen en primer lugar el desplegable "Categoría". 2.- El usuario escogerá, dentro de ese desplegable, la categoría de los productos de los que desee hacer la búsqueda.
Excepciones	Las categorías que se pueden escoger son limitadas. Para Amazon: Portátiles, electrónica, tablets y monitores; Para Mediamarkt: Portátiles, impresoras, monitores, televisores, smartwatches y tablets; Para Tripadvisor: Restaurantes y visitas turísticas. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.9.- Caso de uso selección categoría

Identificador	CU-3
Nombre	Incluir anuncios
Actores	Usuario
Requisitos	RF-1.8, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar si se desea incluir anuncios en la búsqueda.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas.
Flujo principal	1.- El usuario debe marcar la casilla "Incluir anuncios" si desea que estos se cuenten como elementos en la búsqueda. La casilla aparece desmarcada por defecto.
Excepciones	Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.10.- Caso de uso incluir anuncios

Identificador	CU-4
Nombre	Selección ciudad
Actores	Usuario
Requisitos	RF-1.2, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Seleccionar la ciudad de la página web decidida sobre la que se extraerá el conjunto de datos.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web deseada.
Flujo principal	1.- Una vez el usuario ha seleccionado la página web que desea, el apartado "Parámetros de la web" se actualizará mostrando todos los parámetros disponibles para esa web. En la web de Tripadvisor aparecerá el parámetro "Ciudad". 2.- El usuario escogerá, dentro de ese desplegable, la ciudad en la que se realizará la búsqueda de los productos.
Excepciones	Las ciudades que se pueden escoger son limitadas, y son Oviedo, Gijón, Santiago de Compostela, Santander y Bilbao. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.11.- Caso de uso selección ciudad

3.5.1.4- Casos de uso subsistema Selección de elementos

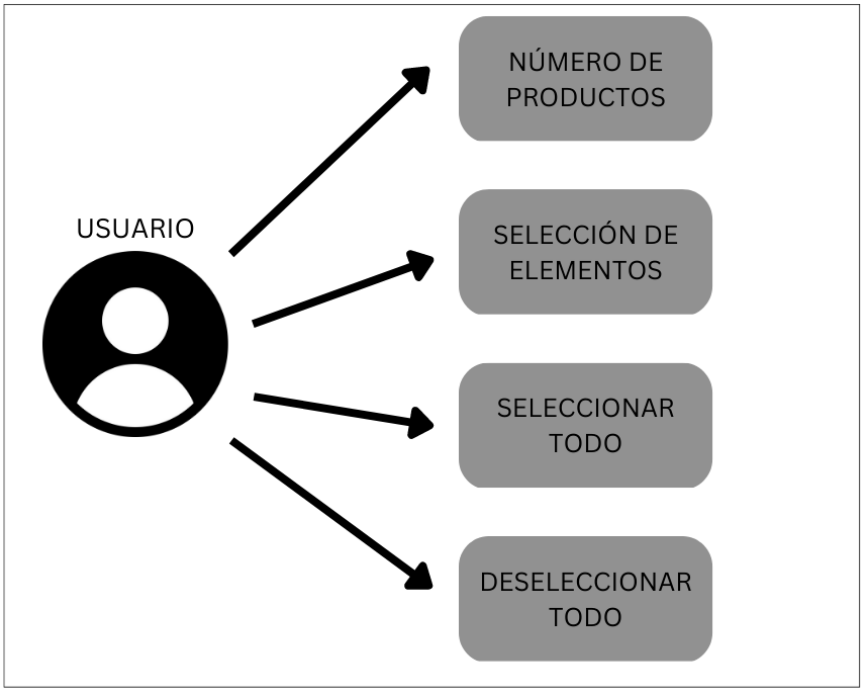

Figura 3.7.- Casos de uso Selección de elementos

Identificador	CU-5
Nombre	Número de productos
Actores	Usuario
Requisitos	RF-1.3, RF-1.4, RF-1.5, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar el número de productos a descargar de la página web.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web y categoría deseadas para que el cuadro de elementos esté actualizado con los disponibles.
Flujo principal	1.- El usuario introduce el valor deseado en la línea de texto "Número de productos".
Excepciones	Si el usuario indica un número de productos mayor del que puede ofrecer la página web, se le indicará y se descargará el máximo posible. Si el usuario no indica ningún número, o introduce el valor 0, se descargarán todos los elementos disponibles. La interfaz no permitirá introducir ningún carácter no numérico en esta línea de texto.

Tabla 3.12.- Caso de uso número de productos

Identificador	CU-6
Nombre	Selección de elementos
Actores	Usuario
Requisitos	RF-1.8, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar los elementos de información de los productos que se incluirán en la búsqueda realizada por la aplicación.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web y categoría deseadas para que el cuadro de elementos esté actualizado con los disponibles.
Flujo principal	1.- Una vez el usuario ha seleccionado la página web y la categoría, el cuadro de "Selección de elementos" se actualizará con todas las opciones posibles. 2.- El usuario seleccionará todos aquellos que desee descargar.
Excepciones	Si el usuario no selecciona ningún elemento, aparecerá un mensaje de error indicando que se debe seleccionar al menos 1 columna y no se le permitirá realizar la búsqueda hasta que seleccione algo. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.13.- Caso de selección de elementos

Identificador	CU-7
Nombre	Seleccionar todo
Actores	Usuario
Requisitos	RF-1.8, RF-1.9, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Seleccionar todos los elementos del cuadro mediante el botón "Seleccionar todo".
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web y categoría deseadas para que el cuadro de elementos esté actualizado con los disponibles.
Flujo principal	1.- El usuario pulsa el botón "Seleccionar todo". 2.- Todos los elementos se seleccionan automáticamente.
Excepciones	-

Tabla 3.14.- Caso de uso seleccionar todo

Identificador	CU-8
Nombre	Deseleccionar todo
Actores	Usuario
Requisitos	RF-1.8, RF-1.9, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Deseleccionar todos los elementos del cuadro mediante el botón "Deseleccionar todo".
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber seleccionado la página web y categoría deseadas para que el cuadro de elementos esté actualizado con los disponibles.
Flujo principal	1.- El usuario pulsa el botón "Deseleccionar todo". 2.- Todos los elementos se deseleccionan automáticamente.
Excepciones	-

Tabla 3.15.- Caso de uso deseleccionar todo

3.5.1.5- Casos de uso subsistema Exportar archivo

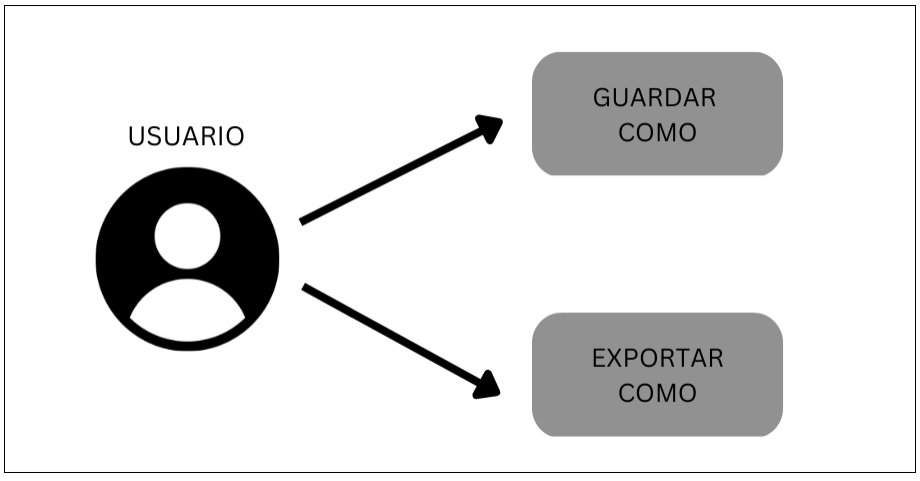

Figura 3.8.- Casos de uso Exportar archivos

Identificador	CU-9
Nombre	Guardar como
Actores	Usuario
Requisitos	RF-1.10, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar el nombre del archivo en el que se exportarán los resultados de la búsqueda.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas.
Flujo principal	1.- El usuario deberá darle un nombre al archivo en el que se exportará toda la información, indicándolo en el cuadro de texto "Guardar como". 2.- Si ese nombre ya está en uso en ese mismo directorio, aparecerá un mensaje de error indicando que debe modificarse el nombre.
Excepciones	Si el usuario escoge un nombre que ya está siendo utilizado, aparecerá un mensaje de error que indicará esto y no permitirá realizar la búsqueda hasta que no seleccione un nombre que no esté en uso. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.16.- Caso de uso guardar como

Identificador	CU-10
Nombre	Exportar como
Actores	Usuario
Requisitos	RF-1.10, RNF-1.1, RNF-1.3, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar el formato del archivo en el que se exportarán los resultados de la búsqueda.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas.
Flujo principal	1.- El usuario deberá seleccionar, a través del desplegable, el formato en el que se exportará el archivo con la información.
Excepciones	Existen 3 opciones de formato de exportación: Csv, Excel y Json. Una vez se pulsa el botón "Buscar" ya no se puede modificar esta selección.

Tabla 3.17.- Caso de uso exportar como

3.5.1.6- Casos de uso subsistema Búsqueda

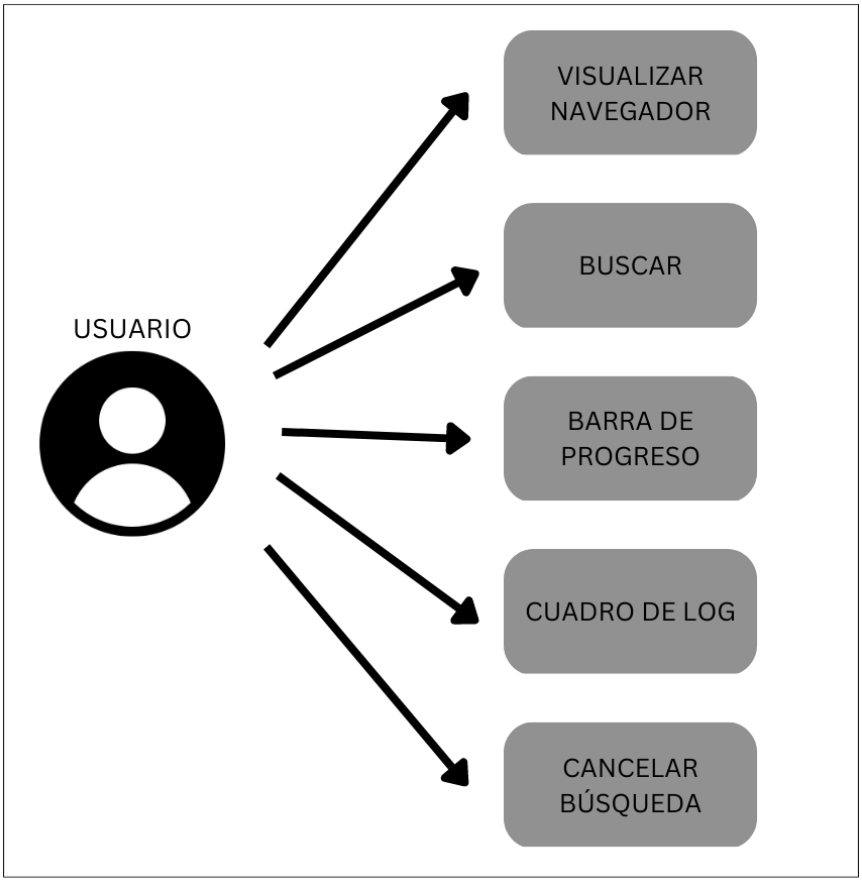

Figura 3.9.- Casos de uso búsqueda

Identificador	CU-11
Nombre	Visualizar navegador
Actores	Usuario
Requisitos	RF-1.13, RNF-1.1, RNF-2.1, RNF-2.3, RNF-2.4
Propósito	Indicar si se desea visualizar o no la ventana del navegador mientras se realiza la búsqueda.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas.
Flujo principal	1.- El usuario podrá marcar la casilla “Visualizar navegador” para indicar si desea que se abra la ventana de navegación durante la búsqueda. En caso de no marcarla, esta ventana permanecerá oculta. La casilla está desmarcada por defecto.
Excepciones	Una vez se pulsa el botón “Buscar” ya no se puede modificar esta selección.

Tabla 3.18.- Caso de uso visualizar navegador

Identificador	CU-12
Nombre	Buscar
Actores	Usuario
Requisitos	RF-1.14, RF-2.4, RNF-1.1, RNF-2.1, RNF-2.2, RNF-2.3, RNF-2.4, RNF-2.5
Propósito	Dar comienzo a la búsqueda una vez se han configurado todos los parámetros para definir el conjunto a descargar y exportar.
Precondiciones	Tener todas las librerías necesarias para la ejecución instaladas. Haber completado todos los campos de la interfaz necesarios para configurar el conjunto de datos que constituirá la búsqueda.
Flujo principal	1.- Una vez el usuario ha configurado completamente todos los parámetros necesarios para realizar la búsqueda, seleccionará el botón "Buscar", que arrancará la búsqueda.
Excepciones	Si no se selecciona ninguna columna en "Selección de elementos", el nombre con el que se guardará el archivo ya existe, o se introduce un carácter no numérico en el "Número de productos" no se permitirá iniciar la búsqueda.

Tabla 3.19.- Caso de uso buscar

Identificador	CU-13
Nombre	Barra de progreso
Actores	Sistema
Requisitos	RF-1.15, RNF-1.1, RNF-2.1, RNF-2.2, RNF-2.3, RNF-2.4, RNF-2.5
Propósito	Informar al usuario del avance de la búsqueda, indicando el % de información que ya se ha descargado.
Precondiciones	-
Flujo principal	1.- Una vez se ha iniciado la búsqueda, el sistema enviará una señal cada vez que se descargue un elemento, lo que actualizará esta barra de progreso y el % que indica el número de elementos descargados. 2.- Cuando finalice la búsqueda, se reseteará y volverá al valor 0.
Excepciones	-

Tabla 3.20.- Caso de uso barra de progreso

Identificador	CU-14
Nombre	Cuadro de log
Actores	Sistema
Requisitos	RF-1.15, RF-1.16, RNF-1.1, RNF-2.1, RNF-2.2, RNF-2.3, RNF-2.4, RNF-2.5
Propósito	Informar al usuario del avance de la búsqueda, indicando los elementos que se van descargando y el número de página en la que se encuentra dentro del sitio web. También indicará el fin de la descarga, el número de elementos descargados y el directorio y formato en el que se exportará el archivo con toda la información.
Precondiciones	-
Flujo principal	1.- Una vez se ha iniciado la búsqueda, el sistema imprimirá un mensaje "Procesando página 1 de #" , siendo # el número de páginas totales que se vayan a recorrer, según la configuración que haya seleccionado el usuario. 2.- Se irá imprimiendo cada elemento que vaya siendo descargado, junto a su descripción. 3.- Cuando sea necesario pasar de página se indicará el cambio. 4.- Una vez se hayan descargado todos los elementos, se indicará con el mensaje: "Fin de la descarga. Se han descargado # elementos", siendo # el número de elementos escogidos por el usuario a descargar. 5.- Seguidamente se indicará el formato en el que han sido exportados los datos junto al directorio en el que será almacenado el archivo, mediante el mensaje "Datos exportados en formato # en la carpeta * .", siendo # el formato seleccionado por el usuario y * el directorio, que será la carpeta correspondiente a la página web escogida, y dentro de esta, la carpeta de la categoría seleccionada.
Excepciones	-

Tabla 3.21.- Caso de uso cuadro de log

Identificador	CU-15
Nombre	Cancelar descarga
Actores	Usuario
Requisitos	RF-1.17, RNF-2.7, RNF-1.1, RNF-2.1, RNF-2.2, RNF-2.3, RNF-2.4, RNF-2.5
Propósito	Interrumpir la búsqueda durante su transcurso a través del botón "Cancelar descarga ".
Precondiciones	Haber iniciado la búsqueda.
Flujo principal	1.- Una vez se ha iniciado la búsqueda, el usuario hará click sobre el botón "Cancelar descarga". 2.- La búsqueda se detendrá y se imprimirá un mensaje en el cuadro de log informando de esto. 3.- Cuando sea necesario pasar de página se indicará el cambio. 4.- Los archivos descargados hasta el momento se guardarán en el archivo indicado en la configuración previa al inicio de la búsqueda.
Excepciones	-

Tabla 3.22.- Caso de uso cancelar descarga

3.6.- DESCRIPCIÓN DEL SISTEMA

3.6.1.- Esquema de la ventana

Para el diseño de la interfaz, se ha priorizado la sencillez para facilitar al usuario el uso y entendimiento de esta, por lo que se colocarán todos los elementos en una única pantalla.

Al iniciar la aplicación aparecerá inmediatamente la pantalla principal, subdivida en zonas, correspondiendo cada una de ellas con los subsistemas en los que se ha dividido el diseño de la aplicación.

El primer subsistema, llamado Selección de web, es el que permite seleccionar el factor más importante: la página web. Está compuesto de un desplegable, que contiene todas las opciones de páginas webs disponibles para la elección. Estas son, por defecto, las páginas webs de Amazon [18], Tripadvisor [19] y Mediamarkt [20].

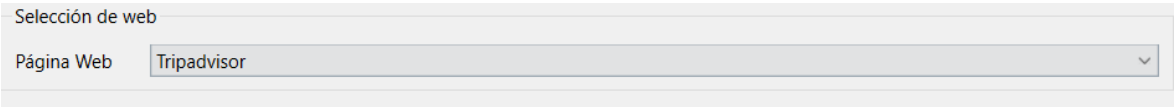

Figura 3.10.- Captura interfaz subsistema selección de web

Seguidamente, está el segundo grupo de elementos: “Parámetros de la web”. Este subsistema es un poco diferente, pues cambiará según la web que se haya seleccionado. Para todas las webs contendrá un desplegable fijo, referente a la categoría de los productos. Al marcar un sitio web en su respectivo desplegable, la lista de categorías se actualizará, de manera que aparezcan las correspondientes opciones para la página web indicada, pues cada una tiene su propia lista de ellas. Además, la web de Tripadvisor contendrá también otro parámetro nuevo, el de ciudad, con su respectivo desplegable.

Por otra parte, también cabe la posibilidad de que, para determinadas webs, el usuario pueda decidir si quiere incluir elemento patrocinados en la búsqueda. Esto se hará a través del desplegable “Incluir anuncios”, teniendo como opciones “Incluir” y “Eliminar”.

Figura 3.11.- Captura interfaz subsistema parámetros de web

Una vez se han seleccionado todos los campos de estos dos subsistemas, el programa ya sabrá con exactitud a qué URL tiene que acceder.

A continuación está el subsistema Selección de elementos, que, como indica su nombre, es el apartado que permitirá al usuario indicar qué elementos de información querrá extraer de cada producto.

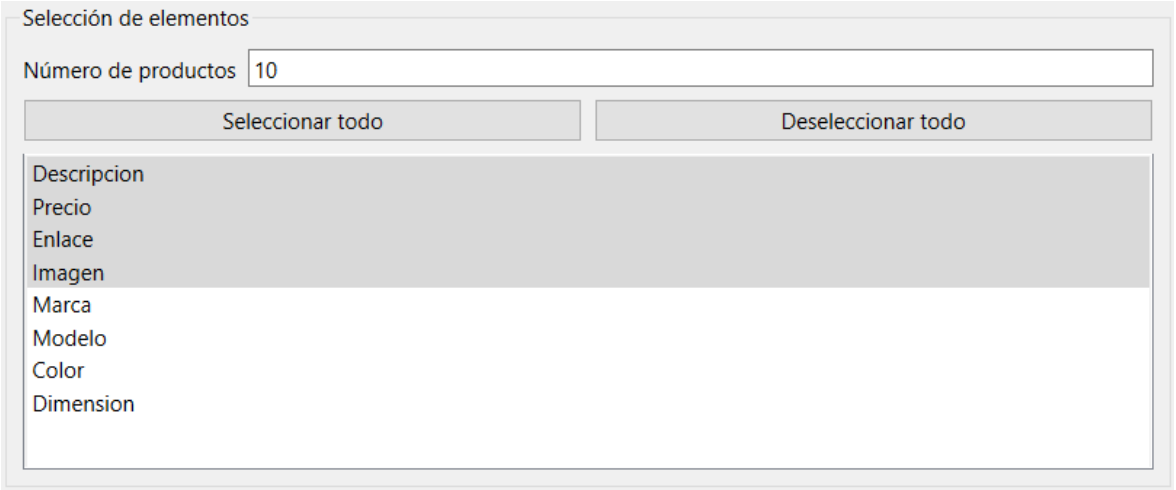

Figura 3.12.- Captura interfaz subsistema selección de elementos

De esta manera, el usuario encontrará primeramente un cuadro de texto que le permitirá indicar el número de productos que querrá obtener del sitio web. Por defecto aparece la cifra 10, pero el usuario podrá modificarlo tantas veces como desee antes de darle al botón "Buscar". Si este cuadro se queda en blanco, o se le da el valor 0, se descargarán todos los elementos de la página.

Además, no se permitirá introducir ningún tipo de carácter que no sea de tipo numérico. Dentro de estos caracteres se ha limitado a los valores de 6 dígitos como máximo, pues ninguna web tendrá un número tan grande de elementos.

Otro de los componentes que definirán la búsqueda es el cuadro que contendrá el listado con todos los elementos de información a extraer de cada producto. Este listado variará según la combinación de página web y categoría que se hayan seleccionado, actualizándose con cada cambio, puesto que cada par web-categoría tiene una lista de elementos característica.

Es obligatorio para la búsqueda que el usuario seleccione, al menos, 1 elementos de la lista. En caso de que no lo haga, al pulsar el botón "Buscar" aparecerá una ventana con un mensaje de error que le indicará que debe seleccionar alguna columna para proceder con la búsqueda.

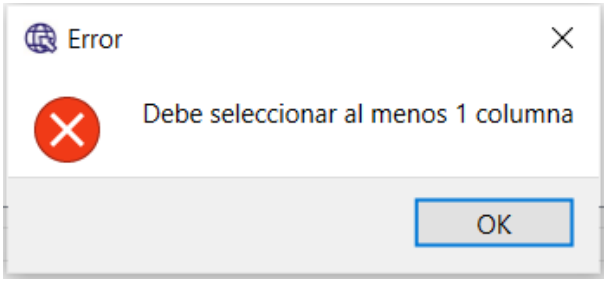

Figura 3.13.- Captura interfaz mensaje de error 1

Por último, este apartado contiene 2 botones, “Seleccionar todo” y “Deseleccionar todo”, que facilitarán al usuario la selección y deselección de todos los elementos de la lista de productos.

Tras este, está el subsistema Exportar archivo, que contiene los pasos necesarios para caracterizar el archivo en el que se exportarán los datos recopilados en la búsqueda. Primeramente, hay un cuadro de texto en el que el usuario podrá indicar el nombre con el que querrá guardar el archivo. Por defecto viene con el nombre “datos” escrito, pero el usuario podrá modificarlo.

Figura 3.14.- Captura interfaz subsistema Exportar archivo

El sistema está diseñado para que no admita 2 archivos con el mismo nombre, por lo que antes de comenzar la búsqueda hará la comprobación de que no exista ningún archivo con la combinación nombre y formato indicado por el usuario en el mismo directorio

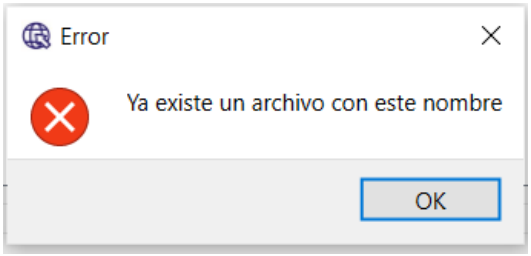

Figura 3.15.- Captura interfaz mensaje de error 2

También hay un desplegable que permitirá escoger el formato en el que será exportado el archivo, teniendo como opciones Json, Csv y Excel. El directorio donde se almacenarán los archivos partirá de la carpeta "out", que, si no existe, se creará con la primera búsqueda. Dentro de esta, habrá una carpeta para cada página web, conteniendo a su vez carpetas individuales para cada categoría.

Por último estará la zona relativa al subsistema "Búsqueda". Primeramente hay una casilla en la que el usuario podrá marcar si desea visualizar el navegador mientras se realiza la búsqueda, estando desmarcada por defecto.

Una vez decidida esta última opción, ya estará completa toda la configuración necesaria para llevar a cabo la búsqueda, por lo que el usuario podrá pulsar sobre el botón "Buscar", que dará comienzo a todo el programa que lleva a cabo la extracción del conjunto de datos personalizado.

Una vez esté en proceso la búsqueda, el usuario podrá obtener información de su desarrollo a través de la barra de progreso y del cuadro de texto informativo. Cuando un producto sea descargado, el sistema enviará una señal que hará que la barra de progreso y el % que está a su derecha se actualicen, hasta llegar al 100% cuando la búsqueda se haya completado. A su vez, aparecerá un mensaje indicando cada ítem descargado junto a la descripción del mismo. Finalmente, se indicará cuando la descarga se haya completado, junto al número de elementos descargados en total y la carpeta en la que serán exportados los datos.

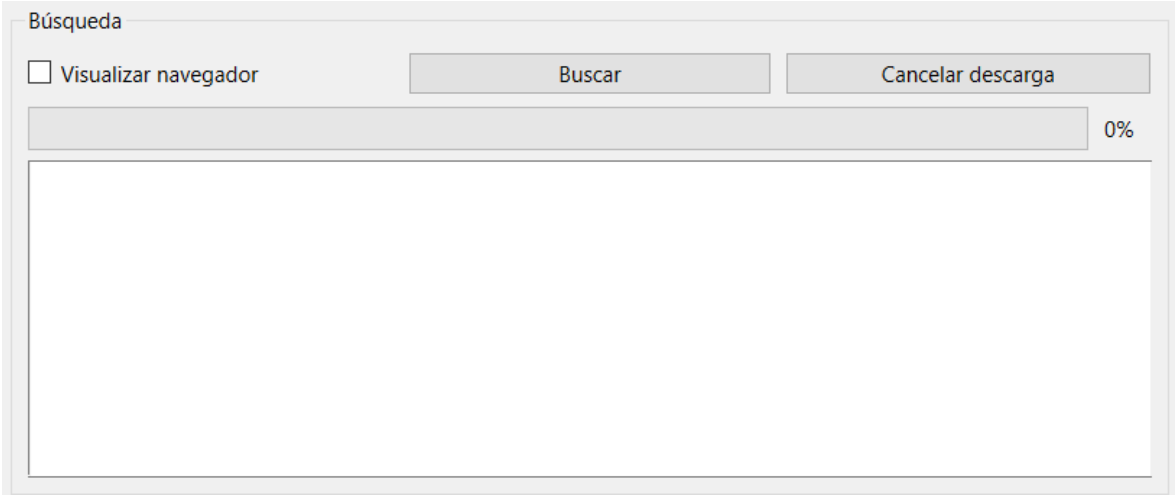

Figura 3.16.- Captura interfaz subsistema Buscar 1

A continuación se muestra una captura de este subsistema cuando se ha finalizado la búsqueda.

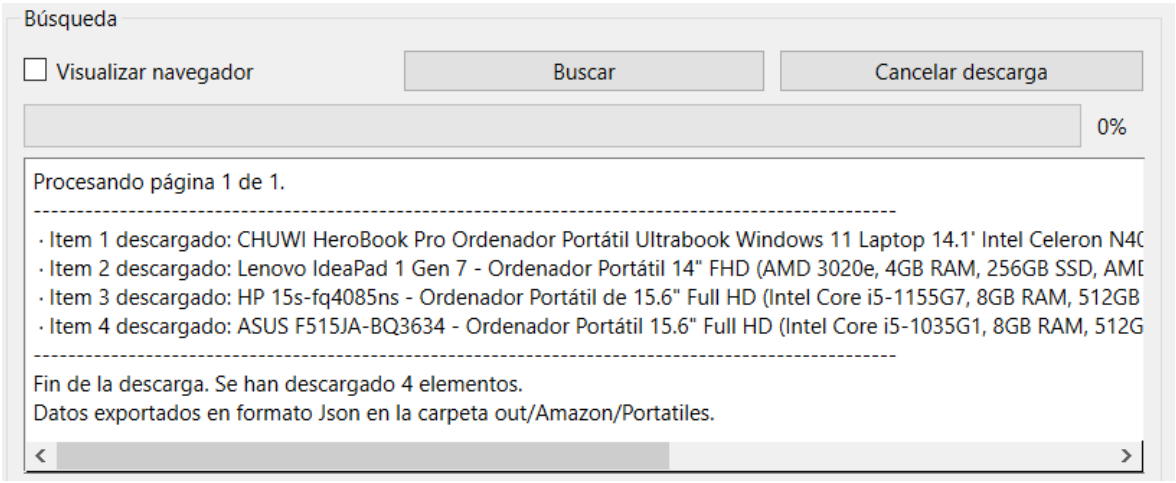

Figura 3.17.- Captura interfaz subsistema Buscar 2

Por último, el usuario también tendrá la opción de cancelar la búsqueda una vez esta haya comenzado. Esta petición se realizará a través del botón “Cancelar descarga”. El proceso se detendrá y se exportarán los datos que se hayan descargado hasta el momento.

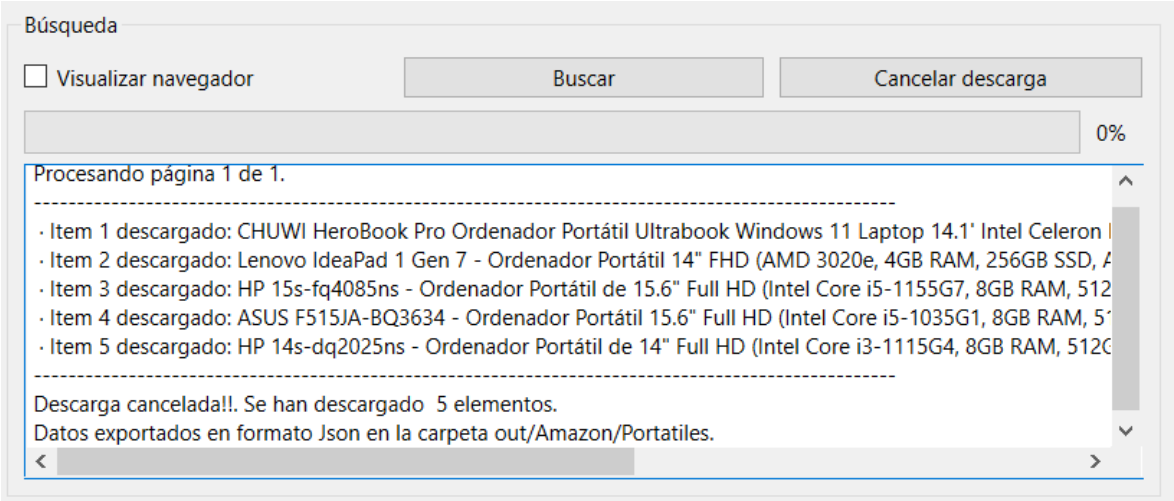

Figura 3.18.- Captura interfaz al cancelar la descarga

Como vista general, en la siguiente figura se incluye una captura que muestra la interfaz gráfica de usuario al completo, uniendo todos los subsistemas mencionados durante este apartado.

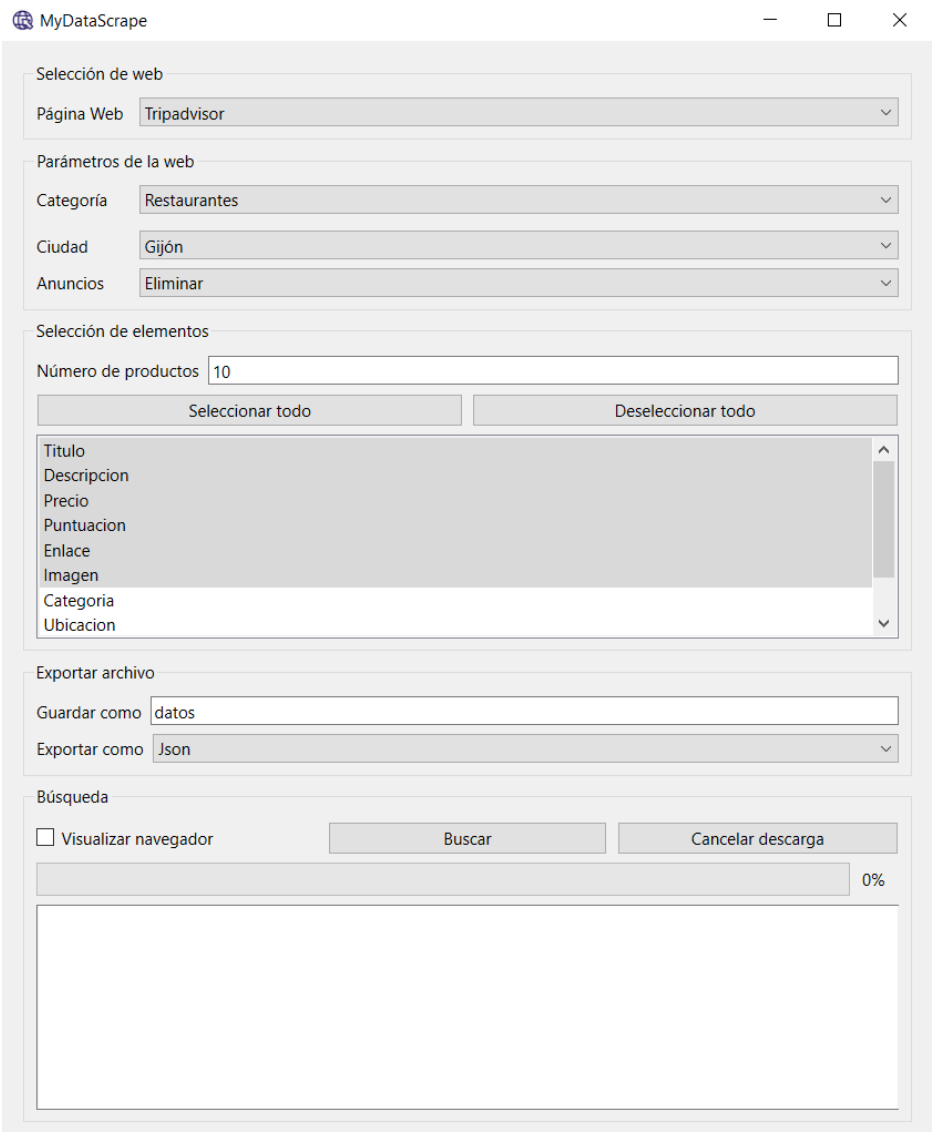

Figura 3.19.- Captura interfaz completa

4. Diseño del Sistema de Información (DSI)

Tras el Análisis del Sistema de Información, se procede al Diseño del Sistema de Información, etapa en la que se definirá la estructura final del sistema y de su interfaz, junto al entorno tecnológico en el que se desarrollará el proyecto.

4.1.- ARQUITECTURA DEL SISTEMA

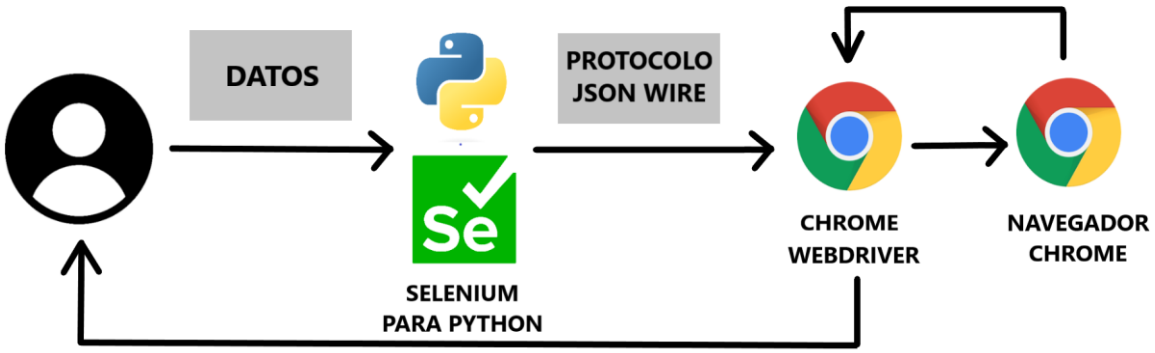

Figura 4.1.- Esquema de la arquitectura del sistema

Una vez el usuario ha configurado en la interfaz todos los factores para definir el conjunto de datos que desea extraer, el sistema da comienzo al proceso del Web Scraping.

La base de esta aplicación reside en la librería Selenium, que permite automatizar la navegación. A través del protocolo WebDriver [24], Selenium enviará peticiones al navegador con el fin de solicitar los datos que el usuario requiere. Para ello, se comunicará con un controlador del propio navegador, en este caso Chrome WebDriver.

Esta comunicación, por su parte, se realiza gracias al protocolo JSON Wire [25], que permite la interacción entre el cliente (Selenium) y el servidor (Navegador) a través de peticiones HTTP.

De esta forma, Selenium envía una petición HTTP con su solicitud y el protocolo JSON Wire se la transfiere al WebDriver del navegador, que lleva a cabo la orden contenida en dicha petición.

Para realizar estas peticiones y que el controlador sepa qué es lo que el cliente le está pidiendo, se utilizan métodos propios de la librería Selenium.

El método principal de este programa es el de localizar los elementos (find_element o find_elements). Permite la localización de cualquier elemento a través de uno de sus atributos HTML. Hay múltiples parámetros a través de los que poder identificarlo, como su ID, su XPath o un CSS Selector, y quedaría a elección del programador seleccionar el más apropiado en cada caso.

También se utilizan otros métodos necesarios, como el de cargar una URL en el navegador (get), o el de cerrar el navegador (close).

Una vez se tienen los datos que se han solicitado, estos se almacenan en un diccionario, que a su vez contendrá tantos diccionarios como número de productos contenga. De esta forma, cada producto individual tendrá su propio diccionario con todos sus atributos.

En este punto entrará en juego otra de las librerías empleadas, Pandas. Esta permitirá convertir el diccionario que contiene todos los datos en un DataFrame. Esta estructura de datos consistirá en un conjunto de filas y columnas, donde cada producto será una fila y cada característica extraída del producto será una columna. Como todos los productos serán procedentes de la misma web, todos compartirán las mismas columnas.

A partir de este DataFrame, que contendrá toda la información recopilada, se seleccionarán únicamente las columnas solicitadas, que serán las que se exportarán a un archivo del formato y nombre indicado por el usuario.

Este archivo se guardará en el directorio cuya raíz es la carpeta “out”, seguidamente la carpeta propia de la página web, y, ahí, en la carpeta de la categoría correspondiente.

4.2.- PATRONES DE DISEÑO

Los patrones de diseño [36] son soluciones a problemas comunes que surgen durante el desarrollo software. Proporcionan soluciones estandarizadas que faciliten un diseño estructurado y más eficiente.

Los patrones de diseño están compuestos por una serie de elementos comunes [37] :

- **Nombre**: Nombre corto que permita identificar y describir el patrón.

- **Problema**: Descripción detallada que defina el problema al que se aplicará el patrón.

- **Solución**: Descripción de cómo el patrón aborda el problema para corregirlo y llegar a los resultados esperados.

- **Ventajas y desventajas**: Lista de los pros y contras que conllevan la aplicación del patrón de diseño.

- **Uso**: Descripción de cómo se aplica y qué resultados se consiguen aplicando el patrón.

Existen varios modelos de patrones de diseño, cada uno adaptado a unas circunstancias y entorno característicos. En este proyecto se ha hecho uso del Modelo-Vista-Controlador (MVC) [38], pues es el patrón más adecuado para esta aplicación, basada en la interfaz gráfica de usuario.

4.2.1.- Modelo-Vista-Controlador

Este es un patrón de diseño de software que, dentro de una aplicación, separa la lógica de negocio, la interacción de los usuarios y la interfaz gráfica en 3 elementos diferentes. Su objetivo es el de conseguir una organización más efectiva que desemboque en una aplicación escalable y que sea sencilla de mantener y mejorar.

De esta manera, los 3 componentes de este patrón son:

1. Modelo: Es el responsable de la funcionalidad de la aplicación y de la ejecución del programa para obtener los datos que serán ofrecidos al usuario.

2. Vista: Es el componente encargado de la interfaz gráfica y de presentarle al usuario los datos recopilados por el programa.

3. Controlador: Es el intermediario de la comunicación entre los otros dos componentes. Recoge las peticiones del usuario, que traduce en acciones al Modelo. A su vez, actualiza la vista para que el usuario reciba el feedback de sus solicitudes.

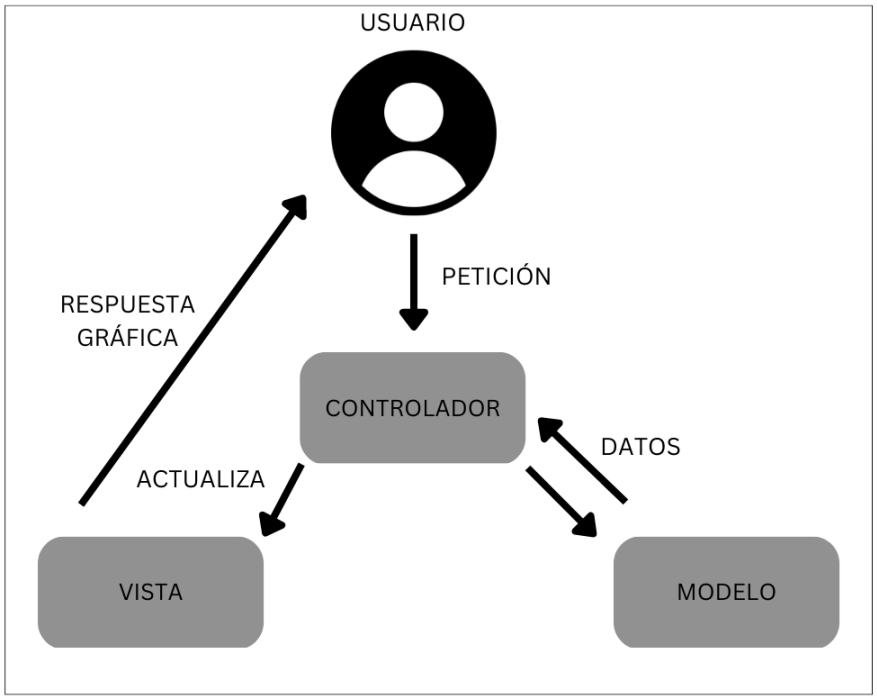

Figura 4.2.- Esquema patrón Modelo-Vista-Controlador

Por tanto, el funcionamiento consistiría en lo siguiente:

1. El usuario interactúa con la interfaz gráfica de usuario para definir el conjunto de datos a descargar.

2. Esta interacción provoca una acción en la vista.

3. El controlador recibe el mensaje de la vista indicando que ha de realizarse una acción.

4. El controlador le envía al modelo la acción que debe realizar, que en este caso será la búsqueda de información.

5. Una vez el modelo se ha actualizado, le devuelve la información al controlador.

6. El controlador genera la actualización sobre la vista.

7. El usuario recibe la respuesta a su interacción.

4.3.- ENTORNO TECNOLÓGICO DE DESARROLLO

En este apartado se indicará el entorno tecnológico en el que se ha desarrollado la aplicación.

4.3.1.- Equipo hardware

Para realizar este proyecto se ha hecho uso de un ordenador portátil con las siguientes características:

- Procesador Intel Core i5
- 12 GB de RAM
- Sistema operativo de 64 bits
- Windows 10

4.3.2.- Equipo software

- PyCharm Community Edition 2021.2.3
- Python 3.10
- QtDesigner

4.3.3.- Tipos de archivos generados

A continuación se detallarán los diversos tipos de archivos que se generarán en este proyecto:

- **Archivos .py**: Son los archivos que contendrán los scripts de Python que implementarán la funcionalidad del programa. Son leídos y ejecutados por el intérprete.

- **Archivos .ui**: Son los archivos que dan forma a la interfaz gráfica de usuario y que son generados por el programa QtDesigner.

- **Archivo .json**: El archivo que contiene toda la configuración relacionada con las páginas webs, categorías y elementos disponibles. La interfaz gráfica lee este archivo y proyecta la información sobre los componentes correspondientes.

- **Archivos .csv, .xlsx, .json**: Son los tres formatos en los que el programa permite exportar el archivo de datos resultante.

    - **.csv**: Es un formato de archivo de texto en el cual los datos están separado por comas y organizados en filas y columnas.

    - **.xlsx**: Es una hoja de cálculo de Microsoft Excel. A partir de este archivo es posible realizar todo tipo de cálculos, tablas o gráficos.

    - **.json**: Este formato de texto representa los datos en la estructura pares clave-valor.

5. Pruebas

A continuación se describe el conjunto de pruebas finales realizadas una vez se ha finalizado la aplicación, con el fin de comprobar que el funcionamiento de cada componente es el esperado. También es importante tener en cuenta que durante todo el proceso de desarrollo del proyecto se han ido realizando pruebas para comprobar el funcionamiento de la aplicación con cada avance.

Se realizarán 2 tipos de pruebas: Test unitarios y pruebas de integración.

5.1.- TEST UNITARIOS

Los test unitarios son pruebas realizadas sobre fragmentos pequeños del código que permiten verificar que su comportamiento sea el correcto.

El proceso para realizar estas pruebas debe ser riguroso y estructurado, puesto que deberán servir como garantía de que todo aquel componente que los pase funciona correctamente y cumple con lo esperado. Este proceso consiste en los siguientes pasos:

1. Identificar el componente a evaluar y cómo debería comportarse.

2. Escribir el código que permita comprobar el funcionamiento del módulo a analizar.

3. Ejecutar el test.

4. Comprobar los resultados y verificar si se corresponden con los esperados.

5. Documentar los resultados y, en caso de que se hayan encontrado errores, hacer las modificaciones precisas y repetir el test.

A continuación se muestra el conjunto de test unitarios mediante los que se ha comprobado el funcionamiento de la aplicación.

Test	Prueba cambio de web.
Requisitos	RF-1.1
Acción desencadenante	Click en desplegable "Página web" y selección de una de las webs disponibles.
Salida esperada	Se cambia la web y se actualiza el apartado "Parámetros de la web", junto con los valores de sus componentes. También se actualiza la lista de elementos de la sección "Selección de elementos".
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.1.- Prueba cambio de web

Test	Prueba cambio de categoría.
Requisitos	RF-1.2
Acción desencadenante	Click en desplegable "Categoría" y selección de las opciones disponibles.
Salida esperada	Se actualiza la lista de elementos de la sección "Selección de elementos" para ofrecer los correspondientes a la combinación Página web-Categoría que haya seleccionado el usuario.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.2.- Prueba cambio de categoría

Test	Prueba seleccionar todos los elementos.
Requisitos	RF-1.9
Acción desencadenante	Click en el botón "Seleccionar todo".
Salida esperada	Se seleccionan todos los elementos que aparezcan en la lista a la vez.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.3.- Prueba seleccionar todos los elementos

Test	Prueba deseleccionar todos los elementos.
Requisitos	RF-1.9
Acción desencadenante	Click en el botón "Deseleccionar todo".
Salida esperada	Todos los elementos de la lista pasan a estar deseleccionados.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.4.- Prueba deseleccionar todos los elementos

Test	Prueba exportar datos en archivo .csv.
Requisitos	RF-1.12, RF-2.1
Acción desencadenante	Click en desplegable "Exportar como", seleccionar la opción "Csv" y darle al botón de "Buscar".
Salida esperada	Se crea un archivo csv en la carpeta out/web/categoría con los datos exportados.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.5.- Prueba datos en archivo .csv

Test	Prueba realizar una búsqueda visualizando el navegador.
Requisitos	RF-1.13
Acción desencadenante	Se marca la casilla "Visualizar navegador" y se pulsa el botón "Buscar".
Salida esperada	Al iniciar la búsqueda se abre una ventana del navegador donde se recorre la web descargando los elementos.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.6.- Prueba realizar una búsqueda visualizando el navegador

Test	Prueba dar un valor negativo al número de productos.
Requisitos	RF-1.3
Acción desencadenante	Escribir valor en la línea de texto "Número de productos" y darle al botón Buscar.
Salida esperada	El programa no permite realizar la búsqueda pues el número de productos no es un valor válido.
Salida obtenida	Error.
Resultado	Error.

Tabla 5.7.- Prueba dar un valor negativo al número de productos

Para resolver el problema planteado en la tabla 5.7 se incluye una comprobación en el código de la aplicación, de manera que solo se permita iniciar la búsqueda una vez se haya verificado que en la línea de texto "Número de productos" hay un valor entero positivo.

Test	Prueba dar un nombre que ya esté en uso al archivo a exportar.
Requisitos	RF-1.9
Acción desencadenante	Escribir nombre en la línea de texto "Guardar como" y darle al botón Buscar.
Salida esperada	El programa no permite realizar la búsqueda pues ya existe un archivo con ese nombre.
Salida obtenida	El programa realiza la búsqueda y el archivo que ya existía se sustituye por el nuevo.
Resultado	Error.

Tabla 5.8.- Prueba dar un nombre que ya esté en uso al archivo a exportar.

El error de la tabla 5.8 se solventa incorporando una comprobación en el código de la aplicación, haciendo que, antes del comienzo de la búsqueda, se chequee si ya existe un archivo con el mismo nombre en el directorio en el que será almacenado. En caso de que sí exista, se indicará mediante un cuadro de diálogo, que persistirá hasta que se elija un nombre en desuso.

Test	Realizar la búsqueda sin seleccionar ningún elemento de la lista para descargar.
Requisitos	RF-1.8
Acción desencadenante	Darle al botón de Buscar sin tener ningún elemento seleccionado de la lista "Selección de elementos".
Salida esperada	El programa no permite realizar la búsqueda pues no se ha especificado un número de elementos.
Salida obtenida	El programa realiza la búsqueda pero el archivo de datos que devuelve está vacío.
Resultado	Error.

Tabla 5.9.- Realizar la búsqueda sin seleccionar ningún elemento de la lista para descargar.

El problema planteado en la tabla 5.9 se soluciona añadiendo otra comprobación más. Esta consistirá en verificar, antes del inicio de la búsqueda, que haya, al menos, 1 columna

seleccionada en la lista de elementos en "Selección de elementos". Del contrario, se abrirá un cuadro de diálogo indicando que será necesario seleccionar, como mínimo, 1 elemento para proceder con la búsqueda.

Test	Realizar una búsqueda solicitando una cantidad de elementos que supere los disponibles en la página web.
Requisitos	RF-1.8
Acción desencadenante	Darle al botón de Buscar introduciendo en la línea de texto "Número de productos" un valor suficientemente grande para suponer que no existirán tantos productos en la página web.
Salida esperada	El programa indica que el valor introducido es superior al disponible y extrae el máximo número de productos.
Salida obtenida	El programa hace la búsqueda hasta que se terminan los productos, cuando lanza una excepción y se cierra.
Resultado	Error.

Tabla 5.10.- Prueba realizar una búsqueda solicitando una cantidad de elementos que supere los disponibles en la página web.

El error indicado en la tabla 5.10 se solucionará comprobando, en cada página web, el número máximo de productos disponibles. Si el valor introducido por el usuario es superior a este, se modifica el número de productos a descargar por el máximo posible, y se informa al usuario.

Test	Cancelar descarga sin tener una búsqueda en curso.
Requisitos	RF-1.17
Acción desencadenante	Darle al botón de Cancelar Búsqueda sin haberle dado previamente al botón de Buscar.
Salida esperada	El programa no hace nada pues no hay ninguna búsqueda en curso que cancelar.
Salida obtenida	El programa lanza un error y se cierra.
Resultado	Error.

Tabla 5.11.- Prueba cancelar descarga sin tener una búsqueda en curso

Para arreglar el error de la tabla 5.11 se añade una comprobación antes de ejecutar la función enlazada al botón Cancelar búsqueda. De esta forma, solo se llevará a cabo la acción una vez el parámetro web, que se crea al iniciar la búsqueda, tenga un valor definido. De esta forma

se podrá verificar que existe una búsqueda ejecutándose sobre la que realizar la acción de interrupción.

5.2.- PRUEBAS DE INTEGRACIÓN

Una vez se ha verificado el funcionamiento de los fragmentos que componen la aplicación a través de los test unitarios, se realiza una prueba de integración. Esta permitirá comprobar el funcionamiento general del programa, de forma que se simule la interacción del usuario con la aplicación.

La prueba consistirá en una agrupación de test unitarios simultáneos, que comprenderán, en su conjunto, todas las partes del programa. De esta forma se podrá verificar al completo el funcionamiento de la aplicación.

Test	Prueba de integración 1.
Requisitos	RF-1.1, RF-1.2, RF-1.3, RF-1.8, RF-1.10, RF-1.12, RF-1.13, RF-1.14, RF-1.15, RF-1.16, RF-2.1
Acción desencadenante	1.- Selección de la web: Amazon 2.- Selección de categoría: Portátiles 3.- Número de productos: 25 4.- Elementos marcados: descripción, precio, marca, modelo y dimensión 5.- Guardar como: datos 6.- Exportar como: Excel (.xlsx) 7.- Casilla “Visualizar navegador” marcada 8.- Hacer click en botón Buscar
Salida esperada	1.- Se abre la ventana del navegador con la página web de portátiles de Amazon cargada. 2.- La barra de progreso y el cuadro de texto se van actualizando con cada producto que descargan. 3.- Una vez la búsqueda ha finalizado, se indica mediante un mensaje de aviso, indicando el directorio donde se encuentra el archivo de datos. 4.- Se accede al archivo “datos.xlsx” en la carpeta out/Amazon/portátiles , el cual contiene la descripción, precio, marca, modelo y dimensión de los 25 primeros portátiles que no sean anuncios de la web.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.12.- Prueba de integración 1

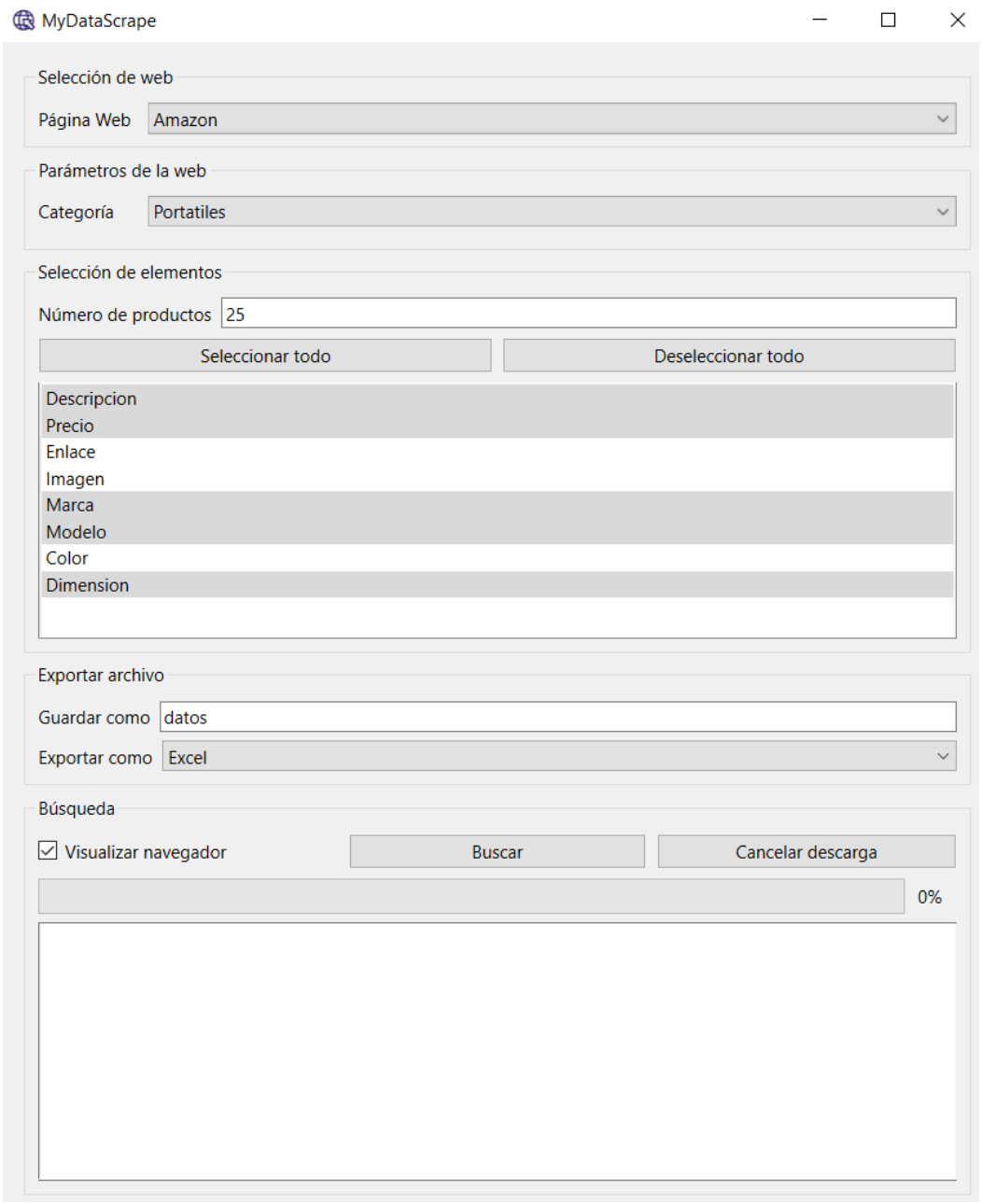

Figura 5.1.- Interfaz con la configuración de la prueba de integración 1

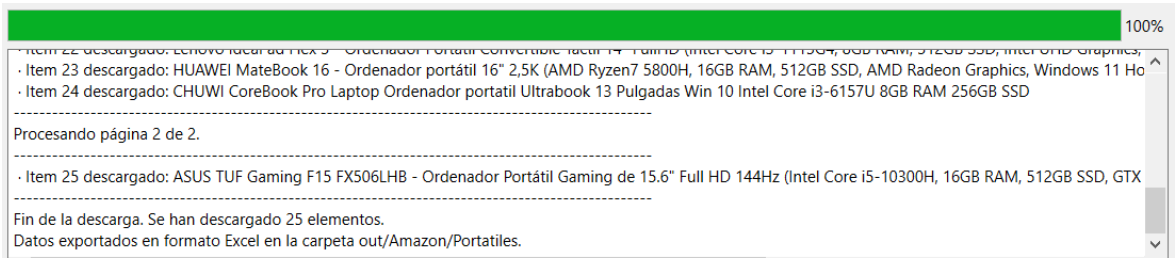

Figura 5.2.- Búsqueda de la prueba de integración 1 finalizada 1

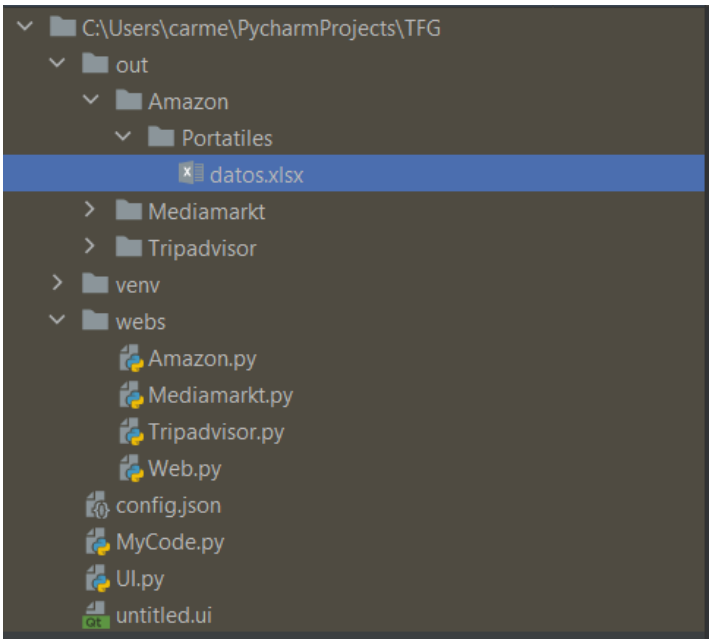

Figura 5.3.- Búsqueda de la prueba de integración 1 finalizada 2

Test	Prueba de integración 2.
Requisitos	RF-1.1, RF-1.2, RF-1.3, RF-1.6, RF-1.7, RF-1.8, RF-1.9, RF-1.10, RF-1.11, RF-1.12, RF-1.13, RF-1.14, RF-1.15, RF-1.16, RF-2.1
Acción desencadenante	1.- Selección de la web: Mediamarkt 2.- Selección de categoría: Televisores 3.- Desplegable "Incluir anuncios" con opción Incluir 4.- Número de productos: Se intenta introducir un texto 5.-El programa no permite introducir caracteres no numéricos. Se introduce la cifra de 10 6.- Hacer click en el botón "Deseleccionar todo" para dejar todos los elementos sin seleccionar 7.- Guardar archivo con el nombre de un archivo que ya existe en ese mismo directorio 8.- Exportar como: Csv (.csv) 8.- Casilla "Visualizar navegador" desactivada 9.- Hacer click en botón Buscar
Salida esperada	1.- Aparece un mensaje de error indicando que ya existe un archivo con ese nombre. 2.- Una vez se ha modificado el nombre para poder continuar, aparece otro mensaje de error indicando que se debe seleccionar al menos 1 columna. 3.- Una vez se selecciona alguna columna de la lista de elementos, se realiza la búsqueda correctamente
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.13.- Prueba de integración 2

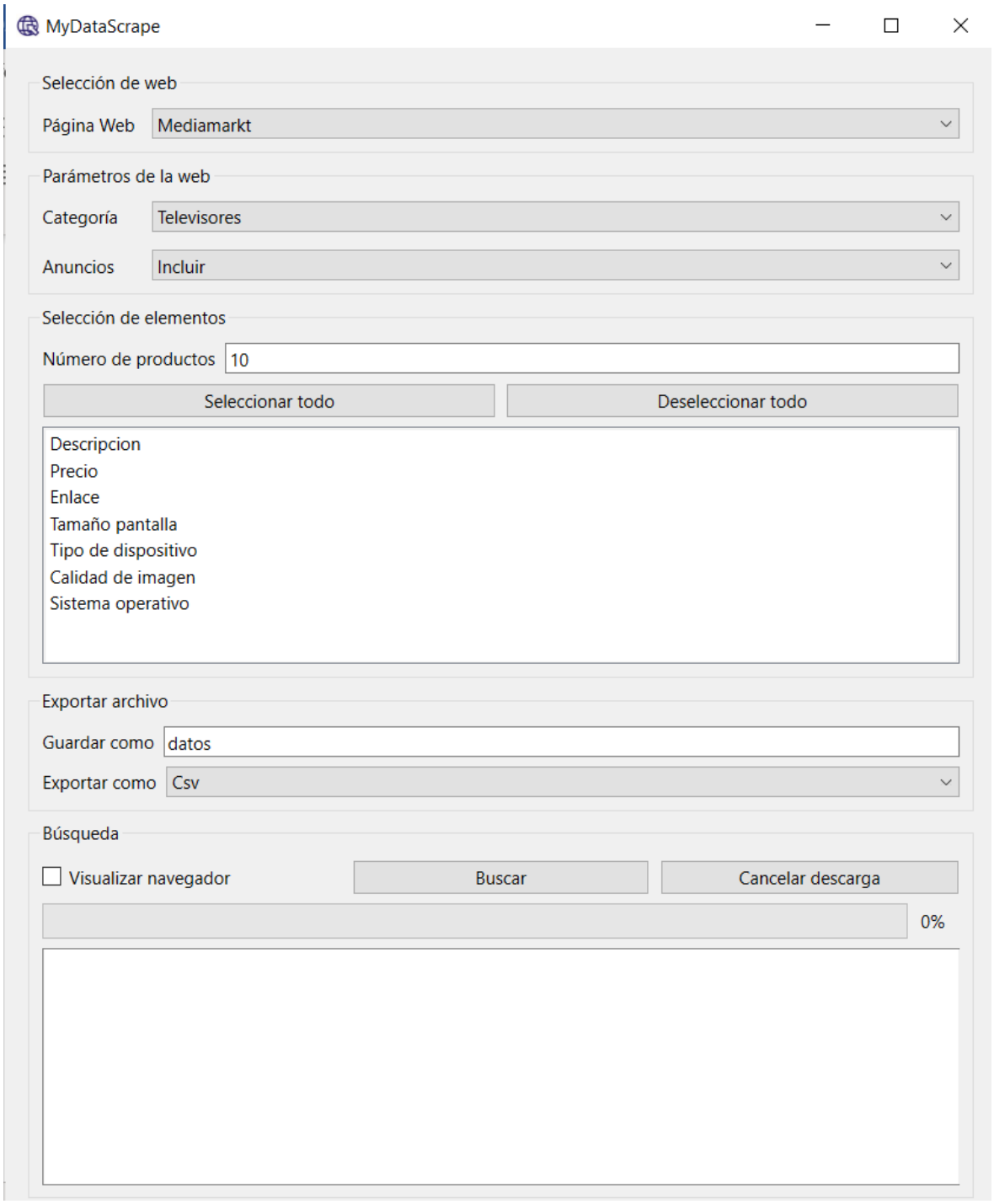

Figura 5.4.- Interfaz con la configuración de la prueba de integración 2

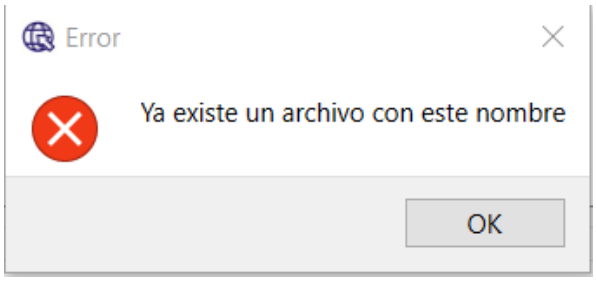

Figura 5.5.- Mensaje de error 1

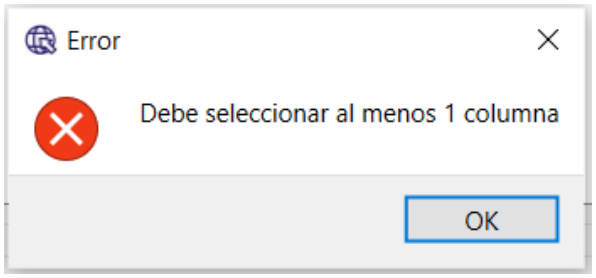

Figura 5.6.- Mensaje de error 2

Test	Prueba de integración 3.
Requisitos	RF-1.1, RF-1.2, RF-1.3, RF-1.7, RF-1.8, RF-1.10, RF-1.12, RF-1.13, RF-1.14, RF-1.15, RF-1.16, RF-1.17, RF-2.7
Acción desencadenante	1.- Selección de la web: Tripadvisor 2.- Selección de categoría: Restaurantes 3.- Selección de ciudad: Gijón 4.- Desplegable “Incluir anuncios” con opción Eliminar 4.- Número de productos: 15 6.- Exportar como datos.json 7.- Hacer click en botón Buscar 8.- Hacer click en botón Cancelar Descarga una vez ya se han empezado a descargar elementos, pero antes de que se finalice la descarga total.
Salida esperada	1.- La descarga se parará. 2.-Aparecerá un mensaje indicando que la descarga ha sido cancelada, el número de elementos descargados y la carpeta a la que se exportarán los datos. 3.- El usuario podrá acceder al archivo con los datos que se habían descargado antes de cancelar la descarga.
Salida obtenida	Igual que la esperada.
Resultado	Correcto.

Tabla 5.14.- Prueba de integración 3

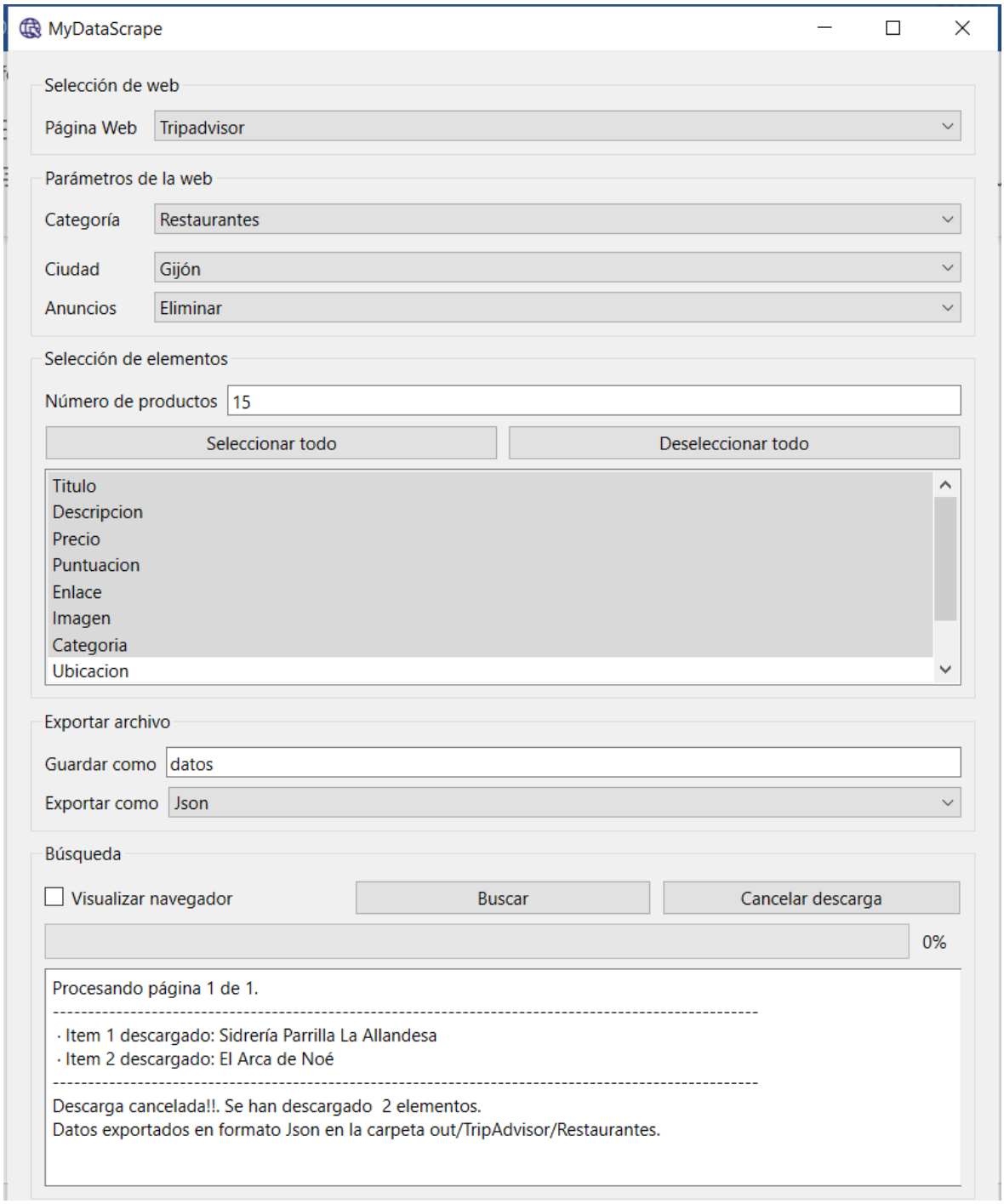

Figura 5.7.-Prueba de integración 3

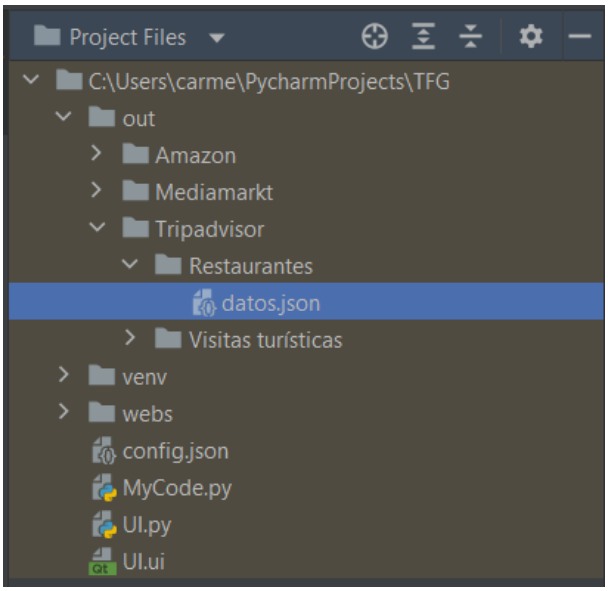

Figura 5.8.-Archivo resultante de prueba de integración 3

6. Manual de usuario

6.1.- USO DE LA APLICACIÓN

Este apartado consistirá en una guía completa para permitir a un usuario nuevo comenzar a utilizar la aplicación desde cero.

El primer paso es la instalación de Python en el equipo en el que se vaya a ejecutar el programa. La versión que se ha utilizado es la 3.10 y, para evitar posibles incompatibilidades, se recomienda descargar esa misma versión.

A continuación se deben instalar todas las librerías de Python que se necesitan para el funcionamiento del proyecto. La figura 6.1. contiene los nombres de todas ellas junto a la versión que se ha utilizado.

Python Package	Version	Python Package	Version
async-generator	1.10	PyQt6-Qt6	6.4.0
attrs	21.4.0	PyQt6-sip	13.4.0
certifi	2021.10.8	PySocks	1.7.1
cffi	1.15.0	python-dateutil	2.8.2
charset-normalizer	2.0.7	python-dotenv	0.21.0
colorama	0.4.6	pytz	2022.7.1
debugpy	1.6.6	requests	2.28.2
et-xmlfile	1.1.0	selenium	4.8.0
exceptiongroup	1.1.0	setuptools	57.0.0
h11	0.14.0	six	1.16.0
idna	3.4	sniffio	1.3.0
numpy	1.24.1	sortedcontainers	2.4.0
openpyxl	3.0.10	tqdm	4.64.1
outcome	1.2.0	trio	0.22.0
packaging	23.0	trio-websocket	0.9.2
pandas	1.5.3	urllib3	1.26.14
pip	21.1.2	webdriver-manager	3.8.5
pycparser	2.21	wheel	0.36.2
PyQt6	6.4.0	wsproto	1.2.0

Figura 6.1.- Librerías y versiones necesarias para la ejecución de la aplicación

A partir de este punto el equipo del usuario ya estaría preparado para ejecutar la aplicación. Para ello existen 2 opciones:

1. Ejecutar desde un editor de código: Toda la aplicación ha sido creada en el entorno de desarrollo PyCharm y es posible ejecutarla desde el mismo. Para ello, habrá que pulsar el botón “Run” , en el menú superior derecho y asegurase de que el archivo que se vaya a ejecutar sea “UI.py”, pues es el que contiene la configuración de la interfaz gráfica de usuario.

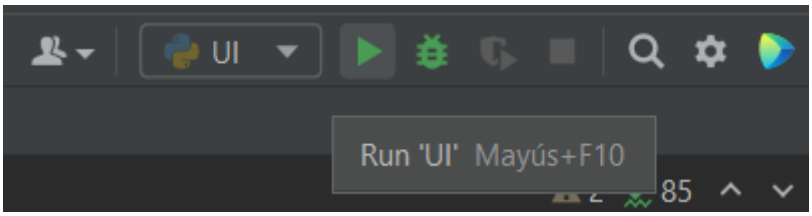

Figura 6.2.- Botón de Run para ejecutar la aplicación en PyCharm

2. Ejecutar desde línea de comandos: También es posible accediendo al directorio donde están alojados todos los archivos que componen el proyecto desde la consola y, una vez ahí, introducir el comando “py UI.py”. De esta forma estaremos indicando que queremos abrir el archivo de Python “UI.py”.

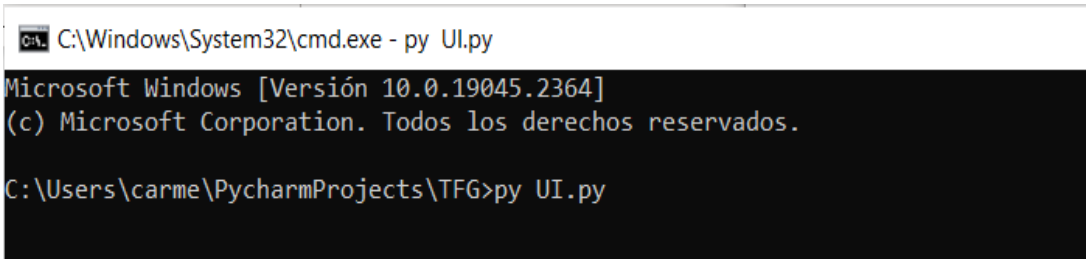

Figura 6.3.- Ejecución de la aplicación desde la línea de comandos

Una vez se ha iniciado la aplicación, el usuario podrá acceder inmediatamente a la ventana principal, donde podrá configurar el conjunto de datos a descargar y, una vez completado, comenzar la búsqueda.

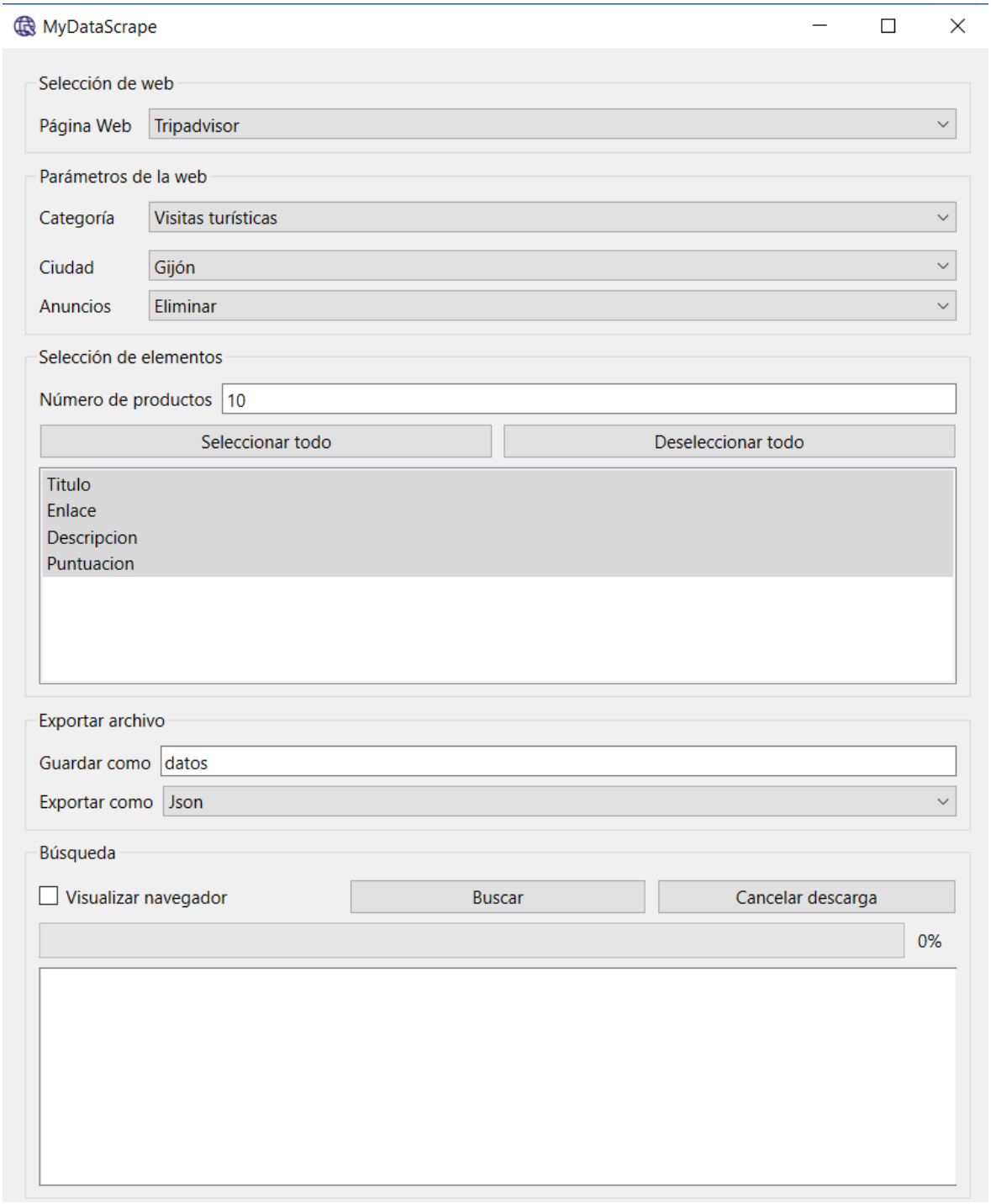

Figura 6.4.- Ventana principal de la aplicación

Una vez se ha finalizado la búsqueda, la propia aplicación creará en su directorio una carpeta llamada “Out”, que será la raíz a partir de la que irán los archivos exportados. A la par que las dos opciones posibles de ejecutar la aplicación, también se podrá acceder a los archivos de resultados de 2 formas:

1. Desde el propio directorio, tanto en el explorador de archivos del equipo como en el del entorno de desarrollo, accediendo a la carpeta out/web/categoría , siendo web y categorías las opciones que haya elegido el usuario al realizar la búsqueda. En caso de que en el entorno de desarrollo no aparezca instantáneamente el archivo, se debe darle al botón de refrescar para que se actualice el directorio.

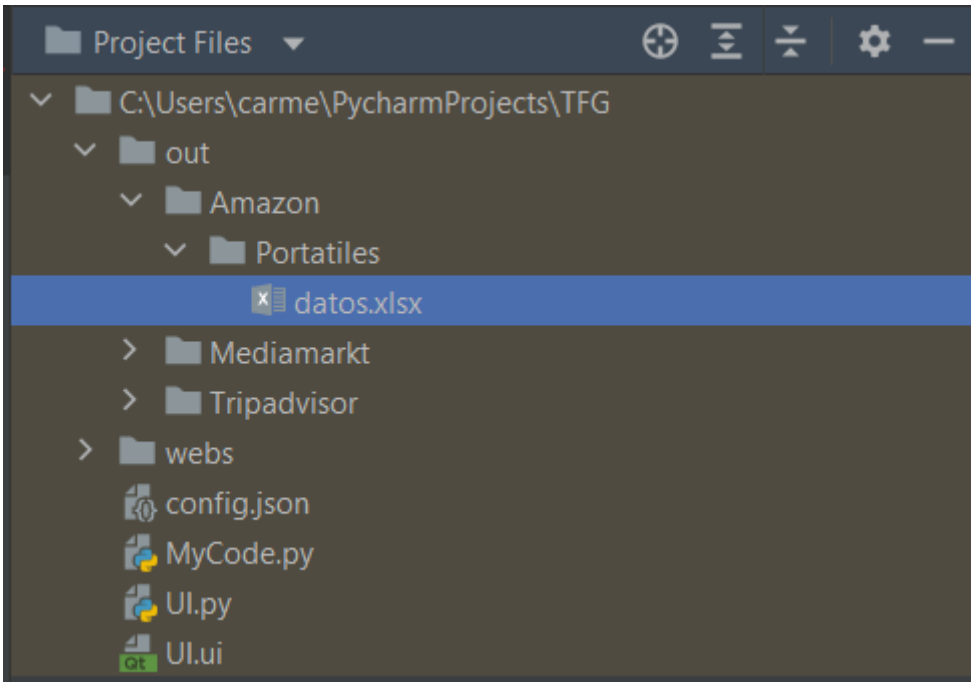

Figura 6.5.- Directorio donde se exportan los archivos

2. Desde la línea de comandos, accediendo al directorio donde se encuentre el archivo e introduciendo como comando su nombre, incluyendo la extensión de formato.

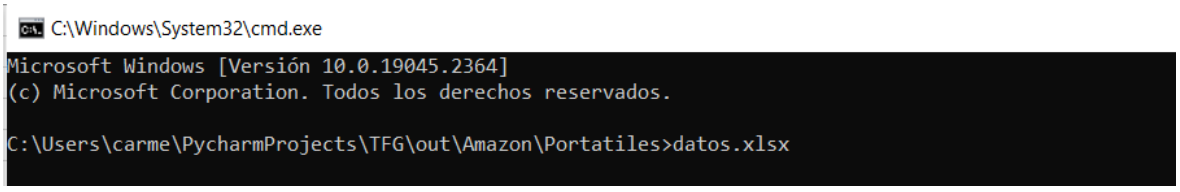

Figura 6.6.- Acceso al directorio desde línea de comandos

6.2.- INTRODUCCIÓN DE NUEVAS PÁGINAS WEBS

A continuación se incluye una guía para indicar al usuario cómo introducir nuevas webs al proyecto, puesto que, aunque se desarrolle con una pequeña muestra, la posibilidad de expansión a un mayor número de sitios webs es uno de los objetivos del proyecto.

Primeramente, una vez se tiene clara la web y los parámetros que esta va a incorporar (como son la categoría o la ciudad), hay que introducirla en el archivo de configuración "config.json".

Dentro de este archivo hay un modelo de ejemplo, llamado "Web de ejemplo", que puede servir como plantilla. La estructura está basada en diccionarios encadenados. De este modo, las claves de la primera capa son las propias webs. Como se puede ver en la figura 6.7., "web de ejemplo" sería la clave y se correspondería con el nombre de la web a introducir.

A continuación, avanzando un nivel más, se crea un nuevo diccionario, cuyas claves serán los parámetros de la página web. En este caso "categories" y "others" serían los ejemplos de muestra, que pueden ser sustituidos por el número de elementos y las opciones que se deseen.

Dentro de esos parámetros, también cabe la opción de que puedan tener varios valores. Aquí es donde se introduce el tercer diccionario, que se correspondería con "Categoría 1", "Categoría 2", "otro_parámetro_1", etc. Estos valores son los que se mostrarán en el desplegable de la interfaz correspondiente al parámetro en cuestión.

Por último, se podrá asignar una lista de elementos a cada opción de parámetro que se haya utilizado en cada caso. Por ejemplo, cada categoría tendrá una lista asociada, cuyos elementos, a su vez, podrán llevar un valor adjunto que permita aportar información sobre ellos.

Siguiendo esta estructura será posible incorporar tantas páginas webs, parámetros y elementos como se desee.

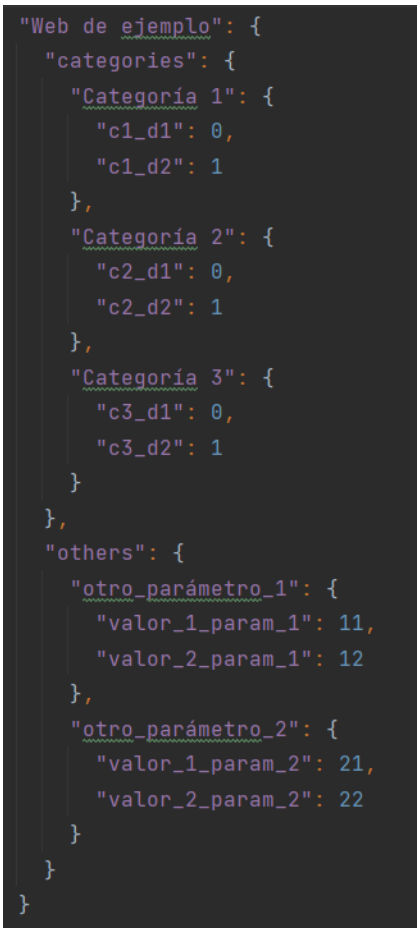

Figura 6.7.- Guía para incorporar una nueva web en config.json

Un detalle de este archivo es que, los elementos a descargar de cada producto están divididos en 2 grupos, y por ellos unos tienen el valor 0 y otros el valor 1. Los elementos con el valor 0 son aquellos que se pueden obtener desde la URL base de la página web, en la vista general de todos los elementos. Sin embargo, los componentes con valor 1 son aquellos que requieren un nivel más de profundidad, y solo pueden obtenerse accediendo a la URL individual de cada producto. Este dato, que separa un grupo de otro, será utilizado dentro de otros de los métodos de la aplicación para indicarle al programa si alguno de los elementos que el usuario ha seleccionado pertenece al segundo grupo.

Una vez hecho esto y comprobado que la nueva web aparece, junto a sus parámetros, en los elementos de selección de la interfaz gráfica, hay que implementarla. Para ello, se creará una clase, dentro de la carpeta “webs”, con el nombre de la nueva web.

Esta clase seguirá la misma estructura que los ya existentes para el resto de webs. Se importarán todos los módulos del resto de scripts y se definirá la clase como herencia de la superclase "Web". Dentro del constructor se incluirán todos los atributos de la clase y se actualizarán sus valores en caso de requerir un valor específico para la nueva página web. Se incorpora también el parámetro "selected_data", que contiene los parámetros seleccionados por el usuario en la interfaz gráfica.

A partir de aquí, tocará adaptar los métodos de la clase Web a la nueva web. Para obtener el número de páginas totales (get_numpags), habrá que comprobar cómo viene indicado este dato en la web. En algunas páginas, el número total se muestra directamente. En otras, solo se proporciona el número total de elementos, por lo que sería necesario hacer el cálculo del número de páginas, siempre y cuando el número de elementos por página se mantenga constante.

El método que dará comienzo a la búsqueda es el método Run. Este comenzará por cargar la URL en el navegador y buscar la forma de identificar a cada elemento, a través de su selector CSS. Para ello hay que hacer una distinción entre los elementos que son anuncios y los que no, pues el usuario tiene la opción de escoger si desea incluirlos o no.

Se hace la comprobación de que el número de elementos pedido por el usuario sea posible dentro de los existentes en la página web. En caso negativo, como se ha indicado en los requisitos de la aplicación, se le informará y se descargará el máximo número de elementos posible.

Una vez se conoce el número de elementos a descargar y el número de páginas a recorrer, se da comienzo a la extracción de los datos para cada item identificado. Esta se llevará a cabo en la función "extract_items_info", en la que se profundizará más adelante.

Durante la extracción de información, se irá pasando de página cuando sea necesario, informando de esto al usuario a través de un mensaje en el cuadro de log.

Una vez se haya terminado la búsqueda, se resetea la barra de progreso para ponerla a 0 y se informa al usuario de que se ha finalizado la descarga. Se cierra el navegador y se guardan los datos en el directorio que se corresponda con la selección del usuario.

Respecto a la función para extraer la información, se deberá identificar cada elemento que se quiera incorporar mediante el código HTML de la web. Todos estos elementos se incluirán en un diccionario, cuyas claves serán los identificadores de cada producto. Como se ha mencionado antes, existen 2 tipos de elementos, los que únicamente son accesibles a través de la URL del producto individual y los genéricos. En el caso de querer introducir elementos del primer tipo, hay que indicarle al driver que abra una nueva ventana con cada item que se vaya descargando.

Durante la ejecución de esta función, también se llama a la función explicada anteriormente, "item_downloaded", que le envíe señales a la barra de progreso para que esta se vaya actualizando y avanzando. A su vez, también se actualizará el cuadro de log, mostrando al usuario la descripción (o el parámetro que se desee) de cada item que se vaya descargando.

7. Conclusiones

Durante el desarrollo de este proyecto, he podido trabajar con una serie de elementos que, hasta ahora, solo había tratado de forma mucho más superficial, y que me han resultado muy interesantes.

El mundo de los datos es un campo en constante crecimiento y desarrollo, y resulta increíble comprobar la infinidad de posibilidades que se ofrecen. Se pueden realizar todo tipo de estudios y análisis, con una amplia gama de herramientas que proporcionan múltiples funcionalidades para el tratamiento de los datos.

Por otro lado, el lenguaje Python es uno de los más utilizados a día de hoy en el desarrollo software y, durante el grado, no había podido manejarlo tanto como me gustaría, pues únicamente se utiliza en un par de asignaturas. Por ello, desarrollar este proyecto en este lenguaje ha supuesto una enorme fuente de aprendizaje para mí. Además, se trata de un lenguaje muy intuitivo y que ofrece una enorme variedad de posibilidades, a través de todas las librerías de las que dispone.

Respecto a la aplicación, también he podido comprobar el nivel de detalle que hay que prestar para su desarrollo. Siempre hay que tener en cuenta todas las posibilidades, y pensar en todo tipo de problemas que puedan surgir con cada posible acción por parte de un usuario. Por ello hay que ser muy meticuloso y prestar mucha atención para poder abarcar todos los posibles casos. También hay que tener siempre como prioridad al usuario, y realizar, tanto el diseño como las funcionalidades, acordes con poder proporcionar la máxima accesibilidad y facilidad de comprensión.

Una vez terminado el proyecto, me quedo muy satisfecha con el trabajo realizado, pues, si bien siempre es posible mejorar y, como se indica en el siguiente apartado, tengo en mente varias ampliaciones que podrían mejorar la calidad y experiencia de uso de esta aplicación, el objetivo principal del proyecto ha sido cumplido. Además, considero que he aprendido un montón y he adquirido habilidades para desenvolverme a la hora de afrontar los problemas que iban surgiendo durante su desarrollo.

8. Futuras ampliaciones

Durante el desarrollo del proyecto han ido surgiendo ideas respecto a posibles mejoras, modificaciones o añadidos que podrían introducirse en el proyecto. Sin embargo, si bien por falta de tiempo o por quedar fuera del alcance principal de este, no han podido ser ejecutadas.

Sin embargo, es importante tenerlas en cuenta, puesto que este se trata de un proyecto escalable, que podría ampliarse y desarrollar muchas más funcionalidades.

En primer lugar, y siendo uno de los propios objetivos de la aplicación, está la posibilidad de ampliar su capacidad y dar soporte a nuevas páginas webs. Este proyecto está desarrollado para 3 webs junto a varias de sus posibles categorías. De este modo, serviría como muestra de la funcionalidad del programa, pero siempre teniendo en cuenta que existe la opción de ampliar las posibilidades a todos los sitios webs que permitan el uso de las técnicas del Web Scraping. Además, en el manual de usuario del apartado 6 se explica detalladamente cómo se puede introducir una nueva web en el proyecto.

Al igual que se puede ampliar el número de páginas webs, dentro de las mismas, siempre que existan más opciones de categorizar los elementos, podrán introducirse. Por ejemplo, en las webs de Amazon o Mediamarkt, podrían introducirse más categorías de productos, como televisiones, libros o electrodomésticos. En la web de Tripadvisor, también podrían añadirse más ciudades sobre las que se permita realizar la búsqueda.

Como otra opción dentro de esta línea, también podrían extraerse más elementos de cada producto, como las reviews que los usuarios de la página web dejan.

Otra opción que también podría resultar interesante sería la de incluir filtros dentro de la búsqueda. De esta forma el usuario podría entrar más en detalle a la hora de seleccionar los datos que quiera extraer. Por ejemplo, si quiere obtener portátiles de la web de Amazon, podría incluir una restricción a través de la que solo se descarguen aquellos que no superen un precio determinado, o que tengan un mínimo de puntuación marcado por el usuario.

Enfocándonos más en la globalización de la aplicación, también cabría la posibilidad de implementar una opción que permita cambiar de idioma la interfaz gráfica del usuario. De esta manera, se podría realizar la búsqueda, exportar los datos del archivo y mostrar la interfaz en el idioma que se haya seleccionado.

Por último, aunque esto no suponga una mejora respecto a la aplicación actual, en el futuro es importante tener en cuenta que las páginas webs se van modificando. Esto supone que el código HTML sobre el que están desarrolladas puede variar respecto al del momento de desarrollo del proyecto, lo que implica que este ya no servirá para realizar la búsqueda. Sin embargo, esto no significa que sea necesario un gran cambio dentro del código de la aplicación, pues únicamente habrá que modificar la etiqueta mediante la que se localiza cada elemento en el método correspondiente, dentro del script de cada web individual.

También hay que contemplar la posibilidad de que alguna de las bibliotecas utilizadas introduzca cambios con nuevas versiones y haya que adaptar el código respecto a esas modificaciones.

En definitiva, este proyecto es una muestra general de las funcionalidades que pueden llevarse a cabo, teniendo una gran cantidad de opciones y mejoras que añadir para poder profundizar en su desarrollo y permitir al usuario un uso más personalizado.

9. Referencias

1. «Wikipedia» wikipedia.org. (Accedido enero 2023) [enlace]

2. «Amazon» amazon.es [enlace]

3. «¿Qué es la Programación Orientada a Objetos?» profile.es. (Accedido enero 2023) [enlace]

4. «Programación orientada a objetos, OOP» computerweekly.com. (Accedido enero 2023) [enlace]

5. «¿Qué es la Herencia en programación orientada a objetos?» ifgeekthen.nttdata.com. (Accedido enero 2023) [enlace]

6. «Polimorfismo en Programación Orientada a Objetos» desarrolloweb.com. (Accedido enero 2023) [enlace]

7. «Programación Orientada a Objetos – Encapsulación (POO Parte 3) » rjcodeadvance.com. (Accedido enero 2023) [enlace]

8. «¿Qué es la interfaz gráfica de usuario GUI?» i.workana.com. (Accedido enero 2023) [enlace]

9. «¿Qué es el web scraping y para qué se utiliza?» ciberseguridad.com. (Accedido enero 2023) [enlace]

10. «Selenium - Documentation» selenium.dev. (Accedido enero 2023) [enlace]

11. «Scrapy - Documentation» scrapy.org. (Accedido enero 2023) [enlace]

12. «¿Qué es el web scraping?» ionos.es. (Accedido enero 2023) [enlace]

13. «HTML» desarrolloweb.com. (Accedido enero 2023) [enlace]

14. «HTML: Lenguaje de etiquetas de hipertexto» developer.mozilla.org. (Accedido enero 2023) [enlace]

15. «JavaScript HTML DOM» w3schools.com. (Accedido enero 2023) [enlace]

16. «Web Scraping: qué es, legalidad, usos y el porqué de su valor diferencial» blog.datary.io. (Accedido enero 2023) [enlace]

17. «Servicios De Web Scraping: Cómo Comenzó y Qué Sucederá en El Futuro» octoparse.es. (Accedido enero 2023) [enlace]

18. «Octoparse» octoparse.es. (Accedido enero 2023) [enlace]

19. «ParseHub» parsehub.com. (Accedido enero 2023) [enlace]

20. «Tkinter — Interface de Python para Tcl/Tk» docs.python.org. (Accedido enero 2023) [enlace]

21. «QT for Python» qt.io. (Accedido enero 2023) [enlace]

22. «Qt Designer Manual» doc.qt.io. (Accedido enero 2023) [enlace]

23. «Playwright» playwright.dev. (Accedido enero 2023) [enlace]

24. «WebDriver - Selenium» selenium.dev. (Accedido enero 2023) [enlace]

25. «JSON Wire Protocol Specification» selenium.dev. (Accedido enero 2023) [enlace]

26. «Arquitectura Selenium WebDriver» tutorialselenium.com. (Accedido enero 2023) [enlace]

27. «Dask» dask.org. (Accedido enero 2023) [enlace]

28. «NumPy» numpy.org. (Accedido enero 2023) [enlace]

29. «Pandas» pandas.pydata.org. (Accedido enero 2023) [enlace]

30. «Series vs. DataFrame in Pandas» educative.io. (Accedido enero 2023) [enlace]

31. «Python - Pandas» w3resource.com. (Accedido enero 2023) [enlace]

32. «Fnac» fnac.es. [enlace]

33. «Milanuncios» milanuncios.com. [enlace]

34. «Tripadvisor» tripadvisor.es. [enlace]

35. «Mediamarkt» mediamarkt.es. [enlace]

36. «¿Qué son los patrones de diseño?» craft-code.com. [enlace]

37. «Qué son los patrones de diseño y por qué utilizarlos» torresburriel.com. [enlace]

38. «Qué es MVC» desarrolloweb.com. [enlace]

Anexo 1: Explicación de las partes principales del código

En este anexo se entrará en detalle en el código base de la aplicación y los métodos usados para obtener cada elemento y exportar los archivos de datos.

ANEXO 1.1.- ESTRUCTURA DEL PROYECTO

En la figura Anexo.1 se muestra la organización de los archivos que componen el proyecto.

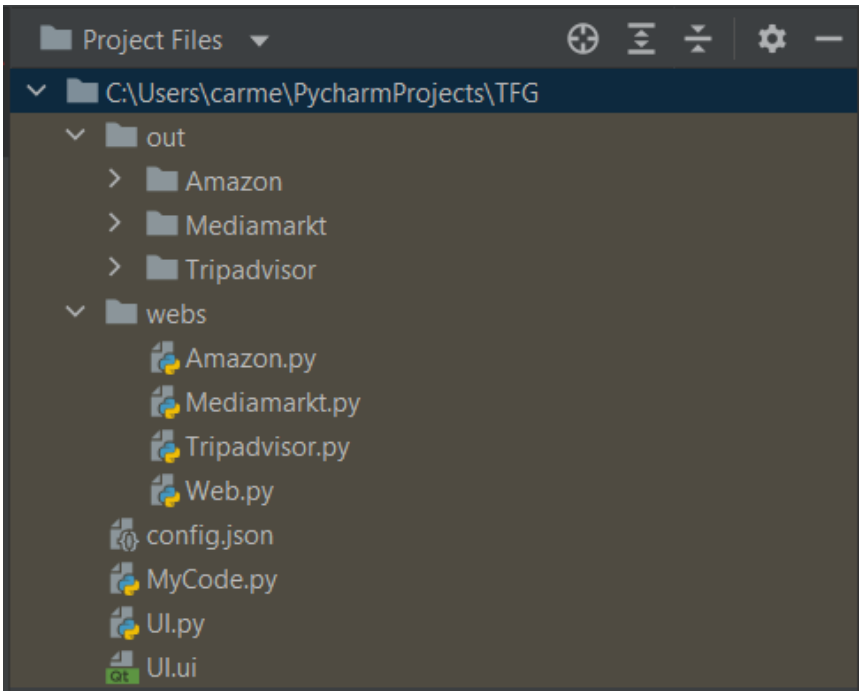

Figura Anexo.1.- Estructura archivos del proyecto

Primeramente se encuentra la carpeta “out”, que es la que se ha marcado como raíz a partir de la que guardarán los archivos resultantes de las búsquedas que realice el usuario. Esta carpeta no es una parte de las que componen el proyecto, sino que es la propia aplicación la

que la genera durante el primer uso por parte del usuario en un equipo nuevo. Una vez está creada, será utilizada para el resto de búsquedas.

Dentro de esta carpeta, se irán creando, a medida que se vayan realizando búsquedas, las carpetas individuales de cada página web y, dentro de las mismas, las de las categorías de productos existentes. De esta forma se tendrá control de qué conjunto de datos contiene cada archivo almacenado.

A continuación, está la carpeta "webs", que contiene los scripts de cada una de las páginas webs que forman parte de la aplicación y la clase "Web", la clase base a partir de la que se crean las clases individuales para cada sitio web.

La clase Web contiene los métodos que heredan las clases de las propias páginas para realizar acciones genéricas, como guardar los archivos de datos en el formato seleccionado por el usuario o actualizar la barra de progreso a medida que se descarguen los datos.

El script de cada web contiene los métodos para obtener los elementos a descargar, cuya forma de localización varía según el sitio web, motivo por el que deben hacerse separadamente.

Seguidamente está el archivo "config.json". Este es un archivo que contiene toda la configuración referente a las páginas webs, sus parámetros y los elementos a descargar. Este archivo será a través del que la interfaz gráfica obtendrá la información que muestra al usuario en los desplegables y listas de elementos que la componen.

El archivo "MyCode.py" es el que lee el archivo de configuración y contiene los métodos para organizar la información de este, de manera que el programa pueda acceder a sus datos ordenadamente.

El script "UI.py" es el archivo que se ejecutará para inicializar la aplicación. Contiene todos los métodos que se utilizan para traducir las acciones que el usuario realiza en la interfaz gráfica en las funciones que debe realizar el programa.

Por último está el archivo “UI.ui” , que es el archivo creado con el programa QtDesigner una vez se ha diseñado la interfaz de forma gráfica, colocando todos los elementos requeridos.

ANEXO 1.2.- EXPLICACIÓN DE UI.py

Como se ha introducido en el apartado anterior, el script “UI.py” es el que contiene la inicialización de la aplicación.

```
if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    ui = MainWindow()
    ui.show()
    sys.exit(app.exec())
```

Figura Anexo.2.- Código “UI.py” parte 1

La ventana principal se crea a partir del archivo “UI.ui”, que contiene todo el diseño de la interfaz. Primeramente, se crean las variables “web_params” y “web”, que se usarán posteriormente.

A continuación, se crea un objeto de la clase MyCode, a través del cual se obtendrá la información para rellenar los objetos de tipo “ComboBox”, que se corresponden con los desplegables de la interfaz de usuario.

También se configuran las conexiones de las acciones que se llevan a cabo cuando se pulsa un botón o se cambia la opción seleccionada en uno de los menús desplegables.

Por último, se crea la restricción que se aplicará a la línea de texto correspondiente a “Número de productos”, de forma que solo se permita introducir valores numéricos de hasta 6 dígitos.

```python
class MainWindow(QtWidgets.QMainWindow):
    # Esta es la clase que nos permite modificar la interfaz

    def __init__(self, *args, **kwargs):
        super(MainWindow, self).__init__(*args, **kwargs)
        uic.loadUi('UI.ui', self)  # Aquí se carga el fichero .ui

        self.web_params = []
        self.web = None

        self.my_code_object = MyCode()
        self.webComboBox.currentTextChanged.connect(self.cambio_web)  # cambiar la web
        self.categoryComboBox.currentTextChanged.connect(self.cambio_cat)  # cambiar la categoria
        self.webComboBox.addItems(self.my_code_object.get_websites())  # desplegable de Webs
        self.datoscomboBox.addItems(self.my_code_object.get_formato()) # desplegable de formato
        self.pushButton.clicked.connect(self.ejecutar_accion)  # botón Buscar
        self.cancelButton.clicked.connect(self.parar_busqueda) # botón Cancelar descarga
        self.selectButton.clicked.connect(self.select_todo)    # botón Seleccionar todo
        self.deselectButton.clicked.connect(self.deselect_todo) # botón Deseleccionar todo
        self.add_web_params()

        self.regexp = QtCore.QRegularExpression("[0-9]\\d{0,5}")  # Restricción para permitir solo números de hasta 6 dígitos
        self.validador = QtGui.QRegularExpressionValidator(self.regexp)
        self.lineEdit.setValidator(self.validador)  # Aplicar restricción sobre lineEdit de N° productos
```

Figura Anexo.3.- Código “UI.py” parte 2

Este proyecto está pensado para soportar una amplia variedad de páginas webs y, ya que estas son muy diferentes entre sí y cada una tendrá sus propios parámetros a configurar, se ha configurado una parte de la interfaz de manera que se añadan los elementos de forma dinámica. De esta forma, se lee qué web está seleccionada en el desplegable correspondiente y se añaden los parámetros que contenga esa web, obteniendo estos del objeto “my_code_object” creado anteriormente. Así se podrá profundizar en las posibilidades que se pueden ofrecer en la aplicación.

Por otro lado, si la interfaz ya está inicializada y se modifica la página web a seleccionar, se eliminarán todos los elementos incorporados en esta área y se sustituirán por los que le correspondan a la nueva página web marcada.

```python
def add_web_params(self):
    # Si ya existen elementos, se eliminan
    for item in self.web_params:
        self.web_params_group.layout().removeWidget(item)
    self.web_params = []

    # Añadir elementos de forma dinámica
    web_params_data = self.my_code_object.get_website_params(self.webComboBox.currentText())
    for param_name in web_params_data:
        item = QWidget()
        item_layout = QHBoxLayout()
        item_layout.setContentsMargins(0, 0, 0, 0)
        item.setLayout(item_layout)

        item.layout().addWidget(QLabel(param_name))

        values = QComboBox()
        values.addItems(web_params_data[param_name])
        item.layout().addWidget(values)

        item.layout().setStretch(0, 1)
        item.layout().setStretch(1, 8)

        self.web_params.append(item)
        self.web_params_group.layout().addWidget(item)
```

Figura Anexo.4.- Código “UI.py” parte 3

En la aplicación, hay elementos que están relacionados entre sí y su valor está influenciado por otros elementos. Para garantizar que estos elementos estén sincronizados y actualizados, es necesario escribir funciones que se ejecuten automáticamente cuando el valor de algún elemento con dependencia cambie. De esta manera, se asegura que todos los elementos relacionados estén siempre en sincronía y actualizados.

En este caso, los elementos a comprobar son el valor de la página web y de la categoría. Cuando cambia el valor de la web, además de actualizarse la parte variable de la interfaz como se ha indicado anteriormente, también cambiarían los posibles valores que toman los parámetros de la web. Por ejemplo, cada página web tiene unas categorías propias, por lo que, según qué página web se haya seleccionado, se mostrarán unos valores u otros de categorías.

El listado de elementos de cada producto también depende de ambas, puesto que cada par web-categoría tiene un conjunto propio de opciones. Por ello, se borrarán los datos que

contiene esa lista, "listWidget" y se llamará a la función "select_datos", que actualizará esa lista para mostrar las elementos de la nueva combinación.

```
    def cambio_web(self):
        web = self.my_code_object.get_website_categories(self.webComboBox.currentText())
        self.categoryComboBox.clear()
        self.categoryComboBox.addItems(list(web))
        self.listWidget.clear()
        self.select_datos()
        self.add_web_params()

    def cambio_cat(self):
        self.listWidget.clear()
        self.select_datos()
```

Figura Anexo.5.- Código "UI.py" parte 4

A continuación se muestra la función "ejecutar_acción", que es la que da inicio al proceso de búsqueda. Primeramente se accede a cada elemento de configuración de la interfaz, almacenando cada valor en su correspondiente variable.

```
def ejecutar_accion(self):
    shownav = self.checkBox.isChecked()
    webpage = self.webComboBox.currentText()
    cat = self.categoryComboBox.currentText()
    numelem = self.lineEdit.text()
    filename = self.archivo.text()
    export_format = self.datoscomboBox.currentText()
    formato = export_format.lower()

    if export_format == "Excel":
        formato = "xlsx"  # En Excel el formato es .xlsx , que es diferente del nombre del formato
    infoextra, selected_data = self.datos_marcados()
```

Figura Anexo.6.- Código "UI.py" parte 5

Posteriormente, se realizan las comprobaciones que se marcan en los requisitos de la aplicación: La comprobación de que el nombre del archivo no esté repetido, que se haya seleccionado alguna columna de elementos y que no se haga uso de la página web que se incluye como ejemplo, y que por ello no está implementada.

En caso de que alguna de estas restricciones no se cumpla, aparecerá un mensaje indicando el problema, que persistirá hasta que se arregle.

Una vez todo esté correcto, se recupera el objeto "web", inicializado con valor nulo al comienzo del programa. Ahora pasará a ser un objeto perteneciente a la clase de la página web que se haya seleccionado, en el que se incluirán como atributos todas las variables que se han definido en el paso anterior.

También se conectará la acción de actualizar tanto la barra de progreso como el cuadro de información a medida que se vayan descargando los elementos. Finalmente, se dará comienzo a la búsqueda.

```
# Mirar si existe un archivo con el nombre escrito
if os.path.isfile(f'out/{webpage}/{cat}/{filename}.{formato}'):
    QMessageBox.critical(self, "Error", "Ya existe un archivo con este nombre", QMessageBox.StandardButton.Ok)
else:
    # Mirar si seleccionó columnas
    if len(selected_data["categories"][cat]) == 0:
        QMessageBox.critical(self, "Error", "Debe seleccionar al menos 1 columna",
                             QMessageBox.StandardButton.Ok)
    else:
        if webpage == 'Amazon':
            self.web = Amazon(category=cat, num_items=numelem, extra_info=infoextra, show_nav=shownav,
                              export_format=export_format, selected_data=selected_data, file_name=filename)
        elif webpage == 'Mediamarkt':
            self.web = Mediamarkt(category=cat, num_items=numelem, extra_info=infoextra, show_nav=shownav,
                                  export_format=export_format, selected_data=selected_data, file_name=filename)
        elif webpage == 'Tripadvisor':
            self.web = Tripadvisor(category=cat, num_items=numelem, extra_info=infoextra, show_nav=shownav,
                                   export_format=export_format, selected_data=selected_data, file_name=filename)
        else:
            QMessageBox.critical(self, "Error", "Esta es una clase de ejemplo y no está implementada",
            QMessageBox.StandardButton.Ok)
            return

        self.web.updateProgress.connect(self.progressBar.setValue)
        self.web.logInfo.connect(self.textEdit.append)
        self.web.start()
```

Figura Anexo.7.- Código "UI.py" parte 6

Las siguientes dos funciones son las que se corresponden con los botones de "Seleccionar todo" y "Deseleccionar todo", que permitirán al usuario marcar o desmarcar todos los elementos de la lista con un solo click.

```
def select_todo(self):
    datos = self.my_code_object.get_website_categories_data(self.webComboBox.currentText(),
                                                             self.categoryComboBox.currentText())
    for i, dato in enumerate(datos):
        self.listWidget.item(i).setSelected(True)

def deselect_todo(self):
    datos = self.my_code_object.get_website_categories_data(self.webComboBox.currentText(),
                                                             self.categoryComboBox.currentText())
    for i, dato in enumerate(datos):
        self.listWidget.item(i).setSelected(False)
```

Figura Anexo.8.- Código “UI.py” parte 7

La función “select_datos” es la encargada de rellenar la lista de elementos, que, como se ha mencionado anteriormente, dependerá de los valores que tomen las variables de la web y la categoría.

```
def select_datos(self):
    selected_website = self.webComboBox.currentText()
    selected_category = self.categoryComboBox.currentText()
    if selected_website != "" and selected_category != "":
        datos = self.my_code_object.get_website_categories_data(self.webComboBox.currentText(),
                                                                self.categoryComboBox.currentText())
        for i, dato in enumerate(datos):
            self.listWidget.insertItem(i, dato)
            if datos[dato] == 0:
                self.listWidget.item(i).setSelected(True)
```

Figura Anexo.9.- Código “UI.py” parte 8

Además de colocar las opciones de datos en la lista, también se necesita un método con el que obtener qué elementos ha seleccionado el usuario. Para ello se crea la función “datos_marcados”.

Gracias a un método propio de la librería Qt, “selectedItems”, se obtienen los elementos de la lista que hayan sido seleccionados, los cuales se añadirán a un nuevo diccionario. Dentro de los posibles elementos a obtener de cada producto, existen dos categorías: los elementos de información que pueden obtenerse desde la vista general de la web y aquellos para los que hay que acceder a cada elemento individualmente.

Por ello, en el archivo de configuración se ha caracterizado cada elemento (los elementos de vista general tienen el valor 0 y a los que se accede de manera individual valor 1). Se comprobará el valor de cada uno de los elementos seleccionados, almacenando en la variable "infoextra" la información de si algún elemento con valor 1 ha sido marcado.

Por último se añadirán también al diccionario el resto de parámetros, junto a sus valores, que tenga la web seleccionada.

```
def datos_marcados(self):

    # Retornar un diccionario con la categoría y otros seleccionados por el usuario para pasarle a la web
    selected_website = self.webComboBox.currentText()
    selected_category = self.categoryComboBox.currentText()
    website_data = self.my_code_object.get_website_data(selected_website)

    return_dict = {"categories": {selected_category: {}}, "others": {}}

    # Añadir los datos de cada item seleccionados al diccionario
    selected_items = [item.text() for item in self.listWidget.selectedItems()]
    infoextra = 0

    for sel_item in selected_items:
        value = website_data["categories"][selected_category][sel_item]
        if value == 1:
            infoextra = 1
        return_dict["categories"][selected_category][sel_item] = value

    # Añadir los otros al diccionario
    for web_param in self.web_params:
        key = web_param.layout().itemAt(0).widget().text()
        value = web_param.layout().itemAt(1).widget().currentText()
        return_dict["others"][key] = {value: website_data["others"][key][value]}

    return infoextra, return_dict
```

Figura Anexo.10.- Código "UI.py" parte 9

Por último, en el caso de que el usuario desee cancelar la descarga y pulse el botón correspondiente, se ejecutará la función "parar_busqueda". Primero, se comprobará que exista una búsqueda en proceso de ejecución, verificando que web tiene un valor no nulo. Si esto se cumple, se llamará a la función Stop, definida en la clase Web.py. Por último, volverá a asignarse el valor nulo al objeto web.

```
def parar_busqueda(self):
    # Por si hacen click al botón antes de iniciar búsqueda
    if self.web!=None:
        self.web.stop()
        self.web = None
```

Figura Anexo.11.- Código "UI.py" parte 10

ANEXO 1.3.- EXPLICACIÓN DE Web.py

Este script aloja la clase base de la que heredarán las clases individuales de cada sitio web. Primeramente, se crea la clase Web, que hereda de la clase QThread. QThread es una clase del módulo PyQt6 que permite la creación de hilos. De esta forma, se estará creando un hilo nuevo, que permitirá el paralelismo de los procesos de la aplicación y mejorará el rendimiento y eficiencia del proyecto. Esto permitirá al usuario acceder a la interfaz gráfica mientras se esté ejecutando el proceso de búsqueda y evitar que se bloquee, problema que ocurriría en el caso de no incluir la clase QThread.

También incorporará las señales que permitan actualizar tanto la barra de progreso como el cuadro de log.

Se creará el constructor de la web, donde se inicializarán todos los atributos necesarios para el desarrollo de la búsqueda con sus correspondientes valores, los desarrollados durante el apartado anterior.

```python
class Web(QThread):
    updateProgress = pyqtSignal(int)
    logInfo = pyqtSignal(str)

    def __init__(self, web_name, url, cookiexpath, num_items, naveg, export_format, selected_data, file_name):
        super().__init__()

        options = Options()
        if naveg == 0:
            options.headless = True  # Para ocultar el navegador
        else:
            options.headless = False  # Se muestra el navegador

        self.web_name = web_name

        self.driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()), options=options)
        self.url = url
        self.cookiexpath = cookiexpath

        self.num_items = num_items  # Número de items que se quieren descargar
        self.downloaded_items = 0  # Número de items ya descargados

        self.export_format = export_format  # Formato en el que se guardarán los datos
        self.selected_data = selected_data  # Columnas que pide el usuario
        self.file_name = file_name  # Nombre del archivo en el que se guardarán los datos
        self.data = {}
```

Figura Anexo.12.- Código “Web.py” parte 1

A continuación se muestran los métodos usados para causar un evento en la interfaz, “item_downloaded”, que actualiza la barra de progreso con cada elemento descargado; “reset_progress”, que resetea y devuelve al valor 0 la barra de progreso una vez se ha terminado el proceso de descarga; “print_l” que imprime el texto de actualización correspondiente en el cuadro de log situado en la parte inferior de la ventana.

```python
    # Eventos de la UI
    def item_downloaded(self):
        self.downloaded_items = self.downloaded_items + 1
        self.updateProgress.emit((self.downloaded_items * 100) // self.num_items)

    def reset_progress(self):
        self.downloaded_items = 0
        self.updateProgress.emit(0)

    def print_l(self, text):
        self.logInfo.emit(text)
```

Figura Anexo.13.- Código “Web.py” parte 2

En esta clase también se definen los métodos usados para realizar la búsqueda de la url en el explorador (get_pag), aceptar las cookies una vez cargada la página (aceptar_cookies), localizar el número total de páginas que componen la web (get_numpags) y guardar el archivo de datos con el nombre y formato indicado, en el directorio correspondiente (guarda_datos). Estos métodos serán usados dentro de las clases de cada web individual, particularizados para cada caso.

Para guardar el archivo, primeramente se crea, en caso de que no exista, el directorio donde se almacenará, el cual será definido en la subclase de cada web. A continuación, se convierte el diccionario que contiene todos los datos descargados en un DataFrame,haciendo uso de la librería Pandas. Dentro de este DataFrame, que contiene la totalidad de datos, se hace un filtrado en el que se dejan solamente las columnas correspondientes a las que haya seleccionado el usuario en la interfaz.

Una vez se tiene esto, se exporta el DataFrame al directorio correspondiente, con el nombre y formato que haya indicado el usuario. Haciendo uso de uno de los métodos definidos en la figura anterior, también se informará al usuario de que el archivo de datos está listo mediante un mensaje en el cuadro de log.

```python
def get_pag(self):
    self.driver.get(self.url)
    time.sleep(5)

def aceptar_cookies(self):
    self.driver.find_element(By.XPATH, self.cookiexpath).click()

def get_numpags(self):
    return 0

def guarda_datos(self, path=""):

    os.makedirs(path, exist_ok=True)

    df = pd.DataFrame.from_dict(self.data, orient='index')
    selected_columns = list(list(self.selected_data["categories"].values())[0].keys())
    df = df[selected_columns]
    if self.export_format == "Excel":
        df.to_excel(f'{path}/{self.file_name}.xlsx')
    if self.export_format == "Json":
        df.to_json(f'{path}/{self.file_name}.json', force_ascii= False)
    if self.export_format == "Csv":
        df.to_csv(f'{path}/{self.file_name}.csv')

    self.print_l(f"Datos exportados en formato {self.export_format} en la carpeta {path}.")
```

Figura Anexo.14.- Código “Web.py” parte 3

Como se ha indicado antes, también existe una función que permita interrumpir el proceso de descarga de datos. Esta comienza cerrando el driver del navegador, reseteando a 0 la barra de progreso e imprimiendo un mensaje indicador de que la descarga se ha cancelado.

Hace uso de la función guarda_datos para exportar los datos que ya se habían descargado en el momento de cancelar la búsqueda. Por último, se llama al método terminate, que finaliza el hilo de ejecución.

```python
def stop(self):
    # Cerrar el navegador
    self.driver.close()

    # Resetear progressbar
    self.reset_progress()

    # Info para el usuario
    self.print_l("-" * 100)
    self.print_l(f"Descarga cancelada!!. Se han descargado  {len(self.data)} elementos.")


    # Guardar los datos en el formato requerido
    self.guarda_datos(path=f"out/{self.web_name}/{self.category}")

    self.terminate()
```

Figura Anexo.15.- Código “Web.py” parte 4

Finalmente, se desarrolla la función “scroll_until_element”, que permite hacer scroll hasta que el driver encuentre un elemento a través de su selector CSS. Este es un método muy útil para la localización de elementos, que será usado en algunas webs.

```python
def scroll_until_element(self, element_css):
    not_end = True
    scroll = 0
    while not_end:
        try:
            a = self.driver.find_element(By.CSS_SELECTOR, element_css)
            not_end = False
        except:
            scroll += 500
            self.driver.execute_script(f"window.scrollTo(0, {scroll})")
```

Figura Anexo.16.- Código “Web.py” parte 5

ANEXO 1.4.- EXPLICACIÓN DE MyCode.py

Primeramente se crea la clase MyCode, cuyo constructor contendrá todos los datos del archivo de configuración “config.json”. También incluirá las opciones que se le proporcionarán al usuario como formatos de exportación del archivo final.

```python
class MyCode:

    def __init__(self):
        with open('config.json', 'r', encoding="UTF-8") as j:
            self.data = json.loads(j.read())

        self.formato = {"Excel", "Json", "Csv"}
```

Figura Anexo.17.- Código "MyCode.py" parte 1

Consecutivamente se definirán todos los métodos necesarios para poder acceder a cada apartado de la información de configuración: Las posibles páginas webs, sus parámetros, sus categorías y los elementos de cada combinación de web y categoría.

```python
def get_websites(self):
    return self.data.keys()

def get_website_data(self, website):
    return self.data[website]

def get_website_params(self, website):
    # Retorna un diccionario con los parámetros de la web y los posibles valores
    return {k: list(self.data[website]["others"][k].keys()) for k in self.data[website]["others"]}

def get_website_categories(self, website):
    return self.data[website]["categories"].keys()

def get_website_categories_data(self, website, category):
    return self.data[website]["categories"][category]

def get_formato(self):
    return self.formato
```

Figura Anexo.18.- Código "MyCode.py" parte 2