ESCUELA INTERNACIONAL DE POSGRADOS

Trabajo Fin de Máster

eip ESCUELA INTERNACIONAL POSGRADOS

Estudiante: Francisco Javier Lázaro Irueste

DNI: 0307096-B

extracción de datos y presentación en aplicación web

Abstract

Proyecto que se enfoca en la creación de una aplicación web con Django en la que tenemos un blog y una herramienta de muestreo de datos scrapeados de diferentes plataformas de venta de enlaces.

En este punto está en el muestreo de los datos de uno de las plataformas, subido una parte del contenido y la posibilidad de registrarse para ver esos datos de la plataforma, que ahora mismo es **Prensarank** también ha podido ser subida la actualización del scrapeo de **Conexo** (este aún no se muestre en una URL de Django), y poder asignar qué entradas son las que te interesa tener almacenadas en tu cuenta.

Resumen

Keywords	Scraper	Django
	Hacking	Datos

Como vemos en esta tabla es un resumen muy bueno de las partes que toca este proyecto que hemos visto a lo largo de este tiempo.

Primero la obtención de datos de otras plataformas, estos datos cómo se va a poder apreciar son varios miles de filas que empezamos a trabajar en csv y luego irán a base de datos.

Esta parte de scraper tiene su lado hacking, ya que tiene que entrar de manera automática, hacer el *login* dentro de la web introduciendo contraseña, nombre de usuario y en algunos casos saltarse el captcha que tiene de seguridad.

Y por último mostrar estos datos en Django ya que el usuario al final usará esta plataforma como interfaz para poder ver estos datos y precios.

Glosario

Los tecnicismos que más dudas nos pueden llevar son:

- **Captcha:** Tipo de medida de seguridad para poder diferenciar entre bots y humanos dentro de una navegación. Esta medida utiliza la pregunta-respuesta como sistema de validación
- **Enlaces:** Son los hipervínculos que están dentro de una página web que dirigen a otra url ya sea interna o externa a la web site.
- **Entradas:** Contenido escrito con imágenes que crearemos para el blog, también se podrá embeber videos de otras plataformas como YouTube.
- **Home:** Sección de inicio en la web que será la de inicio, donde tendremos esta página desde donde podemos ir cualquiera de los puntos de la web principales.
- **Librerías:** Paquetes de código ya programado que podemos importar en nuestras hojas de código ahorrando así tiempo.
- **Link building:** En SEO son las acciones que realizamos para poder conseguir enlaces externos para impulsar la autoridad de nuestra web para los motores de búsqueda. Estos pueden ser de pago o gratuitos.
- **Scraper**: Es el proceso de extracción de información y contenidos en este caso de webs.
- **Script:** Son archivos que contienen el código en este caso de Python y en algunos casos contiene también scripts ejecutables de JavaScript.
- **Tag:** Tags o etiquetas es la forma de marcar nuestras entradas para poder así asignarles un atributo y poder crear con ello incluso filtros por patrones más concretos.

Introducción

La realización de este proyecto creo que es un reto personal a la vez que profesional, ya que toca todos los puntos del máster e incluso algunos que no hemos visto a la vez que es un proyecto que no tiene fin. No tiene fin por varios motivos, subiré una plataforma scrapeada y sus datos pero tenemos varias, estas cambian sus códigos cada X tiempo y además de ello hay muchas y nuevas que van apareciendo, además de esto es la optimización del proyecto, creando diferentes filtros para filtrar los datos y además poder también crear un histórico por ejemplo de precios.

Es por ello que he elegido hacer este proyecto y no hacer uno estándar de los que se propusieron, también para aprovechar de los conocimientos aportados para actualizar y crear mi blog personal.

Objetivos del Trabajo

Este trabajo surge de la necesidad de mejorar y optimizar mi páginas web hecha en WordPress insertando no solo un blog como tal con una página home si no que también la posibilidad de incluir herramientas que me puedan ayudar a monetizar la web, ayudarme a mi marca personal dentro del sector y también poder posicionarla y ampliar con más herramientas que se vayan creando.

Dentro de esta aplicación web se quiere tener un blog hablando de diferentes temáticas de SEO, CRO, Python… Esto ayudará a posicionarnos dentro del tráfico orgánico, con lo que tendremos que ampliar no solo el contenido si no que nos llevará a tener que ampliar la parte del menú de navegación del header.

Además de estos trabajos indirectos que nos llevará, también es uno de los proyectos que por los que ofrecían en la plataforma creo que es más completo y con ello trabajaría las diferentes partes que hemos visto en este curso.

Metodología

He creado los scripts para scrapear de manera independiente, funcionales, teniendo que cambiarlos a un código que evite ciertos errores y puedan cargar de manera automática una vez cada quince días, pudiendo mostrar al usuario la información de los diferentes precios e información de estos enlaces.

A su vez creamos el backend de la web con Django, donde queremos que tengan dos partes muy diferenciadas, la de herramientas y la de web que la utilizaré para hacer de el blog y mi porfolio.

Para este trabajo tengo que crear el esquema de la base de datos ya que tenemos la parte de usuarios, la de los datos de los scrapers y las tablas de post y entradas.

Librerías:

Para este trabajo lo hemos realizado por partes separadas y luego se está implementando todo dentro de un mismo entorno virtual que he creado.

Para ello ha sido necesaria la instalación de varias librerías de Python incluso algunas son de usos temporales y que voy a enumerar las más importantes a continuación.

Django:

La librería del Framework que vamos a utilizar, fue diseñada para que las tareas más comunes sean más rápidas.
Este frame lo hemos elegido porque creo que puede con la capacidad y la envergadura de este trabajo ya que no es únicamente mostrar X datos si no que también hay que trabajar con la lógica que hay detrás para poder mostrarlo además de eso la conexión con diferentes tablas de diferentes bases de datos que vamos a tener.

Fig 1. Imagen de backend de Django. Fuente: backend TFM

Django-ckeditor:

Es una librería que nos ayuda a la edición de texto enriqueciendo sus modelos añadiendo más herramientas para la redacción en el sitio web.

Es una de las librerías que según dice su páginas es una **integración del administrador de Django**

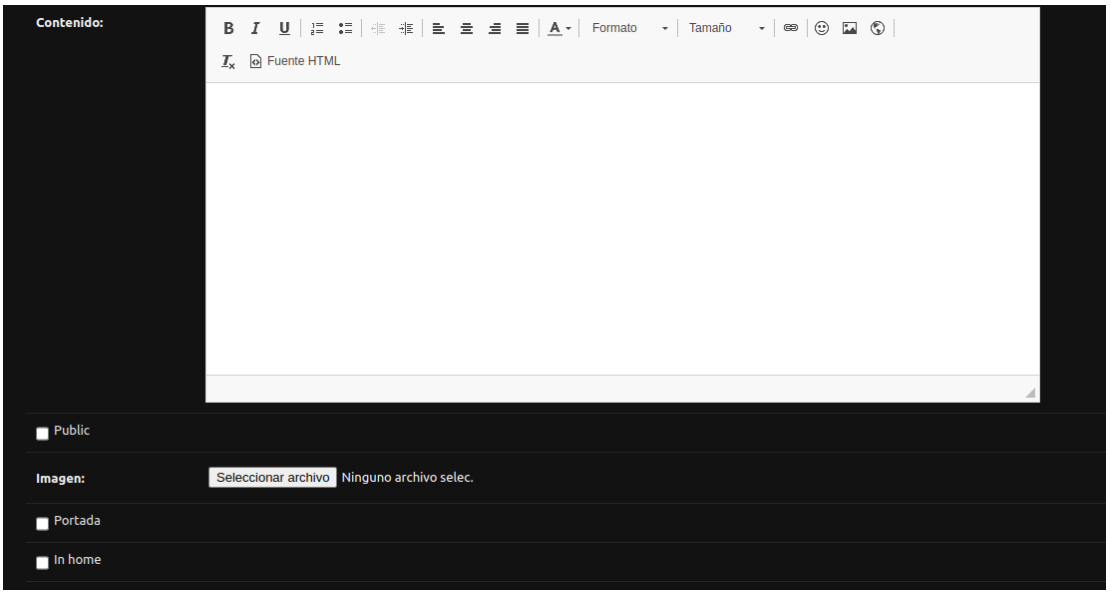

Fig 2. Imagen del backend donde escribimos las entradas. Fuente: backend TFM

En la imagen podemos ver el recuadro donde estamos escribiendo las entradas en este caso. De esta manera una persona con rol de editor no necesitaría tener conocimientos técnicos para poder subir el contenido en la web pudiendo seleccionar la imágen destacada y si va en la portada como destacado en la home o no.

Pillow:

Librería para el procesamiento de imágenes que utilizaremos en los modelos de nuestra web.
Con esta librería podremos manipular, guardar o abrir los distintos formatos que utilicemos en la web.

Psycopg2:

Una librería muy importante, muy útil para la vinculación de nuestro backend con la base de datos que utilizaremos al final.
Django suene venir por defecto con SQLite3 pero es una base de datos que para su uso en pruebas puede estar bien pero no es muy escalable.

Pandas:

Archiconocida librería para el trabajo con datos. Si bien es cierto en nuestro proyecto no es muy necesario en la parte final, me ha servido mucho para poder saber que tal se visualizaban los datos y como estaban siendo descargados sobretodo en la parte de Jupyter.

Jupyter:

Entorno de trabajo para trabajar con Python donde se programa por módulos.
Muy útil en este caso para hacer la configuración de Selenium ya que podía ir por partes y no tenía porqué dejar correr todo el código para saber si funcionaba o no, de esta manera luego lo pasaba a Visual Studio Code que será donde esté albergado todo el código de Django más los scrapers con su introducción a las bases de datos.

```
In [38]: from selenium.webdriver.common.by import By
         import time
         from datetime import datetime,timezone
         from os import name
         from selenium import webdriver
         from selenium.webdriver.support.ui import WebDriverWait
         from selenium.webdriver.support import expected_conditions as EC
         import json
         import requests
         import mysql.connector
         from bs4 import BeautifulSoup
         import sys
         import os
         import pandas as pd
```

```
In [2]: def crear_driver_chrome(headless=True):
            # for windows
            if name == 'nt':
                PATH = r"chromedriver.exe"

            # for mac and linux(here, os.name is 'posix')
            else:
                PATH = r"./chromedriver"

            options = webdriver.ChromeOptions()
            if headless:
                options.add_argument('--no-sandbox')
                options.add_argument('--headless')
                options.add_argument('--disable-gpu')
                options.add_argument('--disable-dev-shm-usage')

            options.add_experimental_option('excludeSwitches', ['enable-logging'])

            options.add_argument('--lang=es')
            options.add_experimental_option('prefs', {'intl.accept_languages': 'es,es_ES'})

            driver = webdriver.Chrome(executable_path=PATH, options=options)

            driver.set_window_size(1920, 1080)
            driver.maximize_window()

            return driver
```

Fig 3. Imagen de Jupyter Notebook con uno de los códigos de scraper de prueba. Fuente Script Prensarank.ipynb

Selenium:

Librería de automatización que también se puede utilizar para la extracción de datos.

En este caso no solo ha sido para este tema si no que hemos necesitado entrar y con un login en ciertas páginas y extraer los datos a través de scripts en JS por optimizar velocidades y esta librería ha sido la mejor para ello.

Aunque he tenido especial cuidado con las excepciones teniendo en cuenta que cuando hacemos un raspado de una web con su paginación cuando llega a la última página tenemos que marcar que ahí no encontrará nada y una de las maneras de conseguirlo ha sido con estas excepciones. No obstante para estar seguros también se ha creado una hoja de logs, para estar seguro de cuándo ha podido fallar y porque cualquiera de los scrapers que se recorran ahora o más adelante con la ampliación de las plataformas que se espera tener listas.

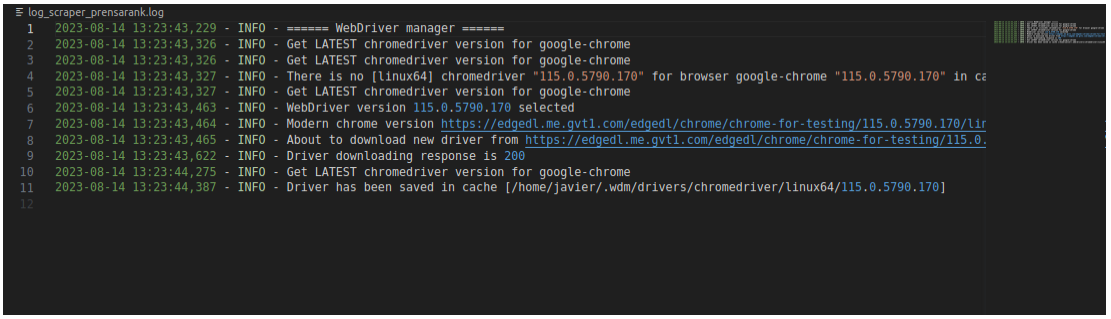

Fig 4. Dato en archivo de logs. Fuente log_scraper_prensarank.log

Carpeta settings

Una de las cosas que he tenido en cuenta para el trabajo en Django ha sido el orden dentro de todas las carpetas que vamos a tener, por lo que he creado una carpeta de aplicaciones donde están albergadas todas las aplicaciones que tenemos y las que necesitemos para un futuro. También he separado en tres la carpeta que viene por defecto en settings.
De de esta manera tener:

Base:

Donde tendremos la configuración principal y que da igual si nuestro proyecto ya está en desarrollo o bien en producción, aquí hay que hacer únicamente las modificaciones que necesitemos de nuestro proyecto que vayan a ser permanentes, como la instalación de aplicaciones o configuración del CKEditor que hemos instalado.

local:

Zona de la configuración que afecta únicamente a al trabajo o código en debug o cuando la web está en desarrollo. En esta parte es donde ahora mismo trabajamos ya que la aplicación aún se encuentra en desarrollo y es donde configuramos los paso que tenemos pero que cuando suban a producción, quiźas por características del servidor haya que modificar.

Producción:

Ahora mismo está vacío ya que no tenemos el proyecto subido y no tenemos por ello las características de este, pudiendo en algunos caso quedarse con las mismas que tenemos en producción, como por ejemplo la base de datos.

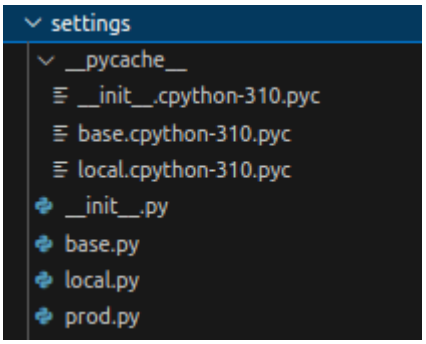

Fig 5: Detalle de carpeta settings del proyecto. Fuente Código TFM

Evaluación de los resultados

Tenemos por una parte la web, donde la estamos montando en Django y por otra parte la parte del scraper donde intentaremos unir estos dos en un mismo proyecto.
La parte de Django más o menos se está solventando en el camino, donde he creado varias aplicaciones:

- **Home:** Será el home de la web, donde tenemos la entrada de la web, digamos que es el escaparate de la web.
- **Entradas:** Será el lugar donde tenemos los *"post"* que escribimos para ampliar los contenidos de nuestra web.

- Users: Tenemos la opción de poder registrarse en la web para llegar a los contenidos que nos aportarán las diferentes herramientas que se van a ir creando.
- **Favoritos:** Los usuarios podrán marcar qué contenido es el que más les interesa y poder así leerlo más adelante
- **Scraper:** Es la primera de varias herramientas que estoy creando, en este caso es un scraper de diferentes precios de plataformas de compras de enlaces. En un primer paso tenemos los scrapers creados en Jupyter teniendo que volcar todo esto a Visual Studio Code para implementarlo dentro del framework de Django.
  Esta parte a día de hoy se está separando para poder tener un control que luego pasará a ser genérico donde tendremos la comparativa de precios de todos estas plataformas.

La idea de este proyecto es que sea escalable, ahora mismo en la muestra siendo el PMV (proyecto mínimo viable) solo tenemos una plataforma, pero la idea es subir los que hemos visto en Jupyter que tenemos en el código y con ello hacer las diferentes bases de datos y comparaciones de precios entre todas ella cuando haya enlaces que sean comunes.

Además de eso tenemos la opción de hacer filtros por el número de tráfico, no solo dame el tráfico X si no intervalos de tráfico entre dos números y filtrar también por el valor de DR (Domain Rating) del dominio.

Estas son las partes que a nivel técnico serán las más interesantes pero da la opción de poder trabajar personas sin conocimientos para subir el contenido gracias a la librería ya mencionada **CKEditor**.

Conclusiones

Django es una herramienta muy potente pero me he dado cuenta que no es únicamente lo técnico a modo de programación si no que la parte de módulos donde dejamos listo la lógica de la base de datos.

Algo que a simple vista se ve bastante fácil pero si queremos hacer filtros, trabajar las diferentes relaciones hay que tenerlas muy en cuenta y con ello y no solo conocer el proyecto si no estas relaciones.

Es un proyecto en el que no se va a quedar aquí ya que la idea es que sea mi web personal y poder seguir ampliando diferentes herramientas que tengo preparadas para mi trabajo y con ello también poder incluso monetizarlo.

Referencias

Estas son las URLs donde he podido encontrar información para poder hacer el trabajo en sí.
Cabe mencionar que la fecha es aproximada a las primeras visitas a estas URLs pero no por ello es la única vez en la que he tenido que acceder a estas páginas para encontrar la información más detallada a cada caso.

Referencia	**Fechas**
PildorasInformáticas: https://www.pildorasinformaticas.es/course/django/	15-may-2023
Django: https://www.djangoproject.com/	19-may-2023
Stack Overflow: https://stackoverflow.com/	19-may-2023
CKEditor: https://django-ckeditor.readthedocs.io/en/latest/#	12-jun-2023
Selenium: https://selenium-python.readthedocs.io/index.html	12-jun-2023
Psycopg: https://www.psycopg.org/	28-jun-2023
Pandas: https://pandas.pydata.org/	28-jun-2023
Foro de Django: https://groups.google.com/forum/#!forum/django-users/home	24-jul-2023

Clasificaciones basadas en view https://ccbv.co.uk/projects/Django/4.2/	19-may-2023

Anexos TFM

Las referencias donde hemos tomado datos para la realización del trabajo han sido:
Adjunto dejo reflejado unos accesos al panel de administrador.
Contraseña: TFM_python03
Nombre: Nexter00
email: prueba@prueba.es

Índice de Figuras