

Documentación módulos

1. Requisitos del Sistema	2
2. Script de Scraping	3
Descripción:	3
Dependencias externas utilizadas:	3
Flujo de ejecución:	3
Estructura de Salida Generada	3
Consideraciones	4
3. Script de Procesamiento	4
Descripción:	4
Dependencias externas utilizadas:	4
Flujo de ejecución:	4
Argumentos Disponibles	4
Ejemplos:	5
Salida del Script	5

1. Requisitos del Sistema

AGREGAR LUCI/RAFA

2. Script de Scraping

Archivo: [scraper.py](#)

Descripción:

Este script automatiza la captura de información desde la plataforma objetivo. Recorre las regiones y ocupaciones, obteniendo el HTML final procesado y extrayendo información relevante para análisis posteriores.

Dependencias externas utilizadas:

- selenium — Para automatizar la interacción con el navegador web.
- beautifulsoup4 — Para analizar el contenido HTML capturado.

Flujo de ejecución:

1. Inicializa el navegador (Chrome/Chromium) con los parámetros necesarios.
2. Accede a la página principal del sistema.
3. Recorre los filtros (regiones, ocupaciones, etc.).
4. Para cada combinación seleccionable:
 - Realiza la selección correspondiente en la interfaz.
 - Captura el HTML final una vez renderizado.
 - Procesa el HTML si corresponde (o se almacena para un análisis posterior).
 - Guarda la salida en carpetas organizadas.

Estructura de Salida Generada

```
/output
  /<region>
    /<ocupacion>
      captura.html
      estado.html
      info.json
```

Archivo	Contenido
captura.html	Representación completa del DOM renderizado.

estado.html	Representación del estado del panel o selector (si aplica).
info.json	Datos limpios y estructurados listos para análisis (texto, métricas, porcentajes, etc).

Consideraciones

- Se recomienda preferir el modo headless para mejorar rendimiento.
 - Las esperas (WebDriverWait) deben garantizar que la interfaz haya cargado.
 - Los nombres de carpetas se sanitizan para evitar errores por caracteres especiales.
-

3. Script de Procesamiento

Archivo: process_htmls.py

Descripción:

Este script toma como entrada las carpetas creadas por scraper.py. Recorre sus contenidos, lee los archivos captura.html y extrae valores relevantes para generar un único archivo CSV consolidado.

Dependencias externas utilizadas:

- beautifulsoup4 — Para extraer información desde el HTML capturado.
- csv o pandas — Para generar la salida final en formato tabular.

Flujo de ejecución:

1. Recibe la carpeta raíz donde se encuentran los archivos HTML.
2. Itera por todas las regiones y ocupaciones.
3. Abre captura.html de cada ocupación.
4. Extrae los datos relevantes mediante BeautifulSoup.
5. Construye un registro por ocupación.
6. Genera un archivo CSV consolidado con toda la información.

Argumentos Disponibles

--root	Carpeta raíz a recorrer (default: outputs/)
--out	Archivo CSV resultante (default: outputs_simple.csv)
--prefix	Filtrar regiones por prefijo (opcional)
--dry-run	No genera CSV, solo muestra los datos

Ejemplos:

```
# Procesar todo y guardar CSV  
python process_htmls.py --root outputs --out tabla_final.csv  
  
# Procesar solo regiones con prefijo  
python process_htmls.py --prefix Bio  
  
# Solo mostrar resultados sin guardar  
python process_htmls.py --dry-run
```

Salida del Script

outputs_simple.csv

```
region  
nombre_ocupacion  
avisos  
vacantes  
profesional  
técnico sup  
media  
basica  
sin_info_educacion  
completa  
parcial  
remota  
otras  
sin_info_modalidad  
salario_nacional_25  
salario_nacional_75  
salarioRegional_25  
salarioRegional_75  
anios_exp_nacional  
anios_expRegional
```