Proyecto integrador - 7Puentes
Desarrollo para Web Scraping

Instrucciones:

1. Formar 3 equipos de trabajo
Podemos asignarlos aleatoriamente o dejarlo a criterio de 7P.

2. Elegir una fuente de datos para scrapear
Por ser el primer proyecto, sugerimos que sea un sitio estático o fácilmente *scrapeable*. Puede ser de medios, de índices de precios como valores de la bolsa, o catálogos de supermercados u otros negocios.
Ejemplos:
- https://www.lanacion.com.ar/
- https://www.boletinoficial.gob.ar/seccion/primera
- Webs con actividades culturales. Ej: https://www.cck.gob.ar/agenda/, https://disfrutemosba.buenosaires.gob.ar/
- Productos: https://www.garbarino.com/ , https://www.mercadolibre.com.ar/
- Ofertas laborales: https://ar.computrabajo.com/, https://www.bumeran.com.ar/

3. Pensar la arquitectura:
- Evaluar limitaciones que pueda tener el sitio a la hora de acceder a los datos.
- ¿Cuándo se va a *scrapear*?
- ¿Qué preprocesamiento hará falta hacer?
- ¿Dónde y en qué formato guardaremos los datos?
- ¿Cómo se actualizará el dashboard?
- ¿Qué podemos hacer si falla el proceso?

Bonus: armar un diagrama del proceso.

4. Crear un repositorio de Github para el proyecto
Debe tener el branch **main** y al menos otro branches intermedio (puede ser **develop**) al cual se harán los merges antes de pasar a main
Bonus: usar un cookiecutter

5. Comenzar el desarrollo en VSCode
Conectar con el repositorio, armar carpetas y estructura básica. Ejemplo:

```
-------
       |- README.md
       |- .gitignore
       /source:
```

```
        |- scraper.py
        |- dashboard.py
/datos:
        |- noticias.csv
```

6. Desarrollar el scraper principal:

Utilizar el debugger de VSCode durante el desarrollo. Usar tipado y documentar las funciones!

Seguir los pasos necesarios:

- Crawl de los links a scrapear con requests/selenium
- Recorrido de los links y extracción de la información con BeautifulSoup o XPATH
- Procesado de los datos
- Guardado de los datos en el .csv

Bonus: Uso de Pandas para procesado

Bonus: Guardado en base local SQL en vez de csv

7. Armado del dashboard

Utilizar plotly y dash para armar un dashboard con algunos gráficos informativos. Deberá actualizarse cuando cada vez que se levanta con el cron, o al ingresar nuevos datos.

8. Hacer un cron con ambos procesos

Scrapear los datos regularmente (una vez por día por ejemplo) y levantar el dashboard en un horario predeterminado.

Bonus: Enviar un mensajito por Telegram con datos o avisando de su ejecución. Consultarnos por Discord para hacer eso :)

Finalización:

Asistiremos a todos los proyectos durante los vivos y de manera asincrónica por Discord.

Adicionalmente hacia la finalización, **al mejor trabajo vamos a proponer una asistencia especial para *deployarlo* en producción con Docker y Cloud, para compartirlo como ejemplo de proyecto de alumnos de Humai.**

Timeline:

Fecha	Checkpoint
20/07	*Propuesta de Proyecto*
25/07	*Estructura base*
29/07	*Scraper armado*
03/08	*Dashboard armado*
05/08	Entrega final funcionando