

Informe del proyecto integrador Desarrollo para Web Scraping

EQUIPO 1



INTEGRANTES

Tomás
Battistella

Adriano
Canavero

Agustina
Chrzanowski

Noelia
Ledesma

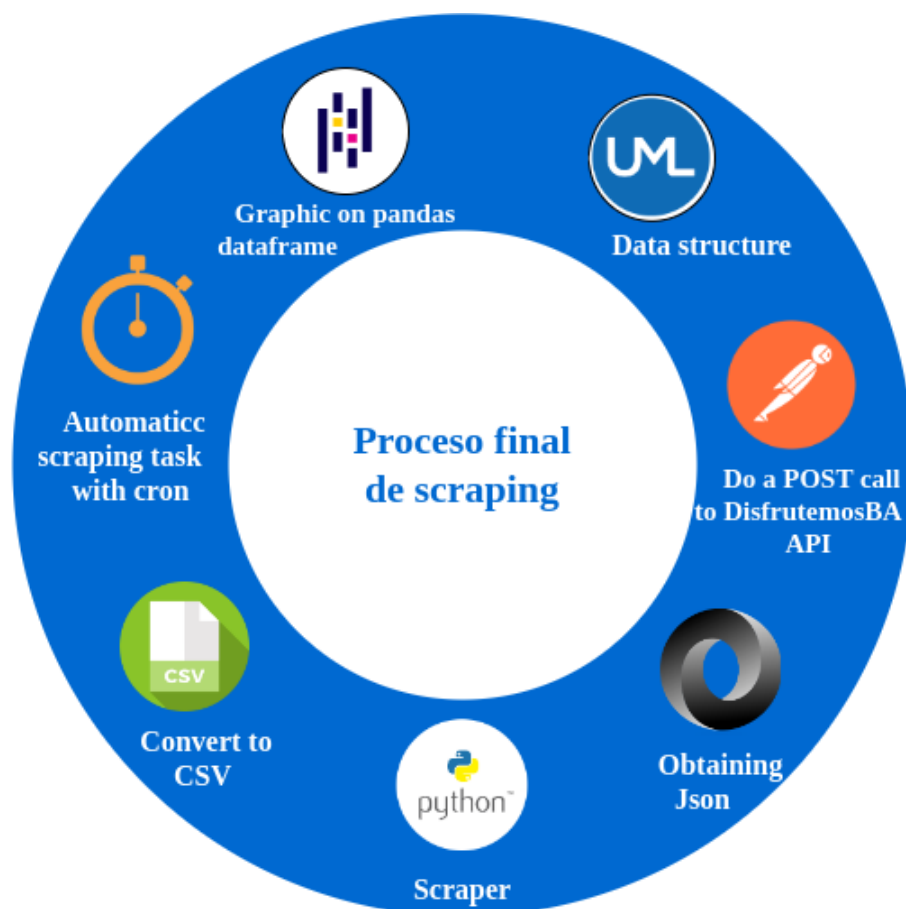
INTRODUCCIÓN

El presente informe se propone presentar las características del proyecto de scraping sobre la página web <https://disfrutemosba.buenosaires.gob.ar/> del Gobierno de la ciudad de Buenos Aires, llevado a cabo en el contexto del trabajo final integrador del curso de webscraping dictado por el intituto Humai y siendo los asistentes a dichas clases integrantes del equipo de web data extraction de la empresa 7puentes.

OBJETIVOS GENERALES

Obtener automáticamente los datos de la totalidad de las actividades culturales que se realizan durante el día en la ciudad de Buenos Aires y representarlos de forma gráfica con un dashboard.

Diagrama del proceso



Primera etapa

Seleccionamos la página a escrapear en base a la buena accesibilidad de los datos, la variedad de campos disponibles para agrupar y visualizar mediante filtros. Antes de decantarnos por esta web descartamos otras opciones por razones de moneda atada a la variación cambiaria, baja frecuencia de cambio en los datos o porque necesitábamos hacer una normalización de los ítems a comparar y no disponíamos de tiempo suficiente para ello.

📍 Algo para hacer cerca tuyo



≡ Disfrutemos BA

🔍 Empezá tu búsqueda

Siempre hay algo para hacer en la Ciudad

¡Celebrá la cultura de 12 países en la Ciudad!

Hoy



Este finde



Esta semana



Este mes



Segunda etapa

Elegimos la estructura de la tabla de datos que queríamos extraer de la fuente de datos. También elegimos el tipo de dato de cada campo en la estructura que tendría una base en SQL en caso de crearla:

Actividades	
PK	<u>Id</u>
	tipo_actividad
	fecha
	hora
	tipo_lugar
	direccion
	gratis
	descripcion
	url

Actividades

Id (INT AI PK NN)
 Tipo actividad [VARCHAR(200) NN]
 Fecha [DATE]
 Hora [TIME]
 Tipo de lugar [VARCHAR(200) NN]
 Dirección [VARCHAR(900)]
 Gratuito [BINARY]
 Descripción [TEXT]
 Url [VARCHAR(250)]

Tercera etapa

Se realizó la búsqueda de dónde estaban alojados los datos de las actividades. Se encontró una api en el cual se encontraba la información de cada actividad en formato JSON. Se analizaron las rutas de acceso a cada dato particular.

The screenshot shows a REST client interface with a POST request to `https://disfrutemosba.com/api/search`. The request body is in JSON format. The response status is 200 OK, and the response body is also in JSON format, showing a list of activities with pagination information.

```

1 {
2   "type_of_experiences": [],
3   "kind_of_places": [],
4   "min_price": 0,
5   "max_price": 3200,
6   "only_free": 0,
7   "dates": [],
8   "moments": [],
9   "districts": [],
10  "only": "",
11  "page": 1
12 }
  
```

The response body is shown in the 'Body' tab, displaying a JSON object with the following structure:

```

{
  "activities": {
    "pagination": {
      "current_page": 1,
      "data": [
        {
          "id": 34,
          "activity_id": 46,
          "activity_type": "App\\Models\\Event",
          "created_at": "2022-02-08T19:53:28.000000Z",
          "updated_at": "2022-02-08T19:53:28.000000Z",
          "activity": {
            "id": 46,
            "type_of_experience_id": 2,
            "admin_id": 6,
  
```

Cuarta etapa

Respondimos a la guía de desarrollo propuesto por el instituto Humai

- **Evaluar limitaciones que pueda tener el sitio a la hora de acceder a los datos.**

La limitación que puede tener nuestro sitio es que para poder traer los datos es necesario hacer una llamada a una api. Además, la otra dificultad es que los horarios en cada una de las actividades tienen formatos diferentes.

- **¿Cuándo se va a *scrapear*?**

Se va a *scrapear* dos veces al día en nuestro horario laboral.

- **¿Qué preprocesamiento hará falta hacer?**

Tendremos que hacer un request POST a la página para conseguir en la respuesta el DOM en formato JSON. Una vez que tengamos ese json tendremos que extraer los datos necesarios para poder parsearlos a un archivo CSV, de esta forma podremos actualizar el dashboard cuando sea necesario y además tener un registro histórico por cada vez que se corre.

- **¿Dónde y en qué formato guardaremos los datos?**

Se guardan en una carpeta en el github del proyecto y en formato CSV.

- **¿Cómo se actualizará el dashboard?**

El dashboard se irá actualizando a medida que los csv se vayan cargando al sistema.

- **¿Qué podemos hacer si falla el proceso?**

Si falla el proceso de bajada de datos se intentará nuevamente y se irá guardando los csv anteriores para tener un respaldo por si acaso.

Quinta etapa

Con la información localizada se construyó el código del scraper y se extrajo un CSV de prueba con éxito. Posteriormente el código se perfeccionó para que fuera idóneo para graficar. Se utilizó pandas, plotly y dash como herramientas de visualización.

```
from calendar import week • Untitled-1 - Proyectos - Visual Studio Code
Terminal Help
from calendar import week
import requests
import lxml.html
import os
from datetime import datetime
import unicodedata

import csv
test as manage_csv

def get_json_page(page_num: int):
    url = 'https://disfrutemosba.com/api/search'

    payload = '{"type_of_experiences\":[],\n"kind_of_places\":[],\n"min_price\":0,\n"max_price\":3200,\n'
    headers = {'Content-Type':'application/json',
               'sec-ch-ua':'Not(A)Brand;v="99", "Google Chrome";v="103", "Chromium";v="103"',
               'Accept':'application/json, text/plain, */*',
               'sec-ch-ua-mobile':'?0',
               'User-Agent':'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome'
               'sec-ch-ua-platform':'Linux'}

    response = requests.request("POST",url, headers=headers,data = payload)

    return response.json()

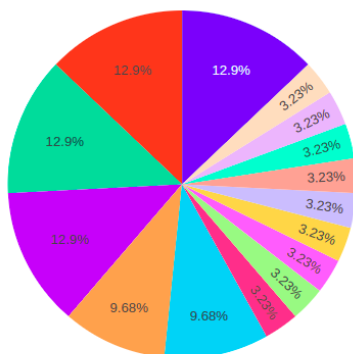
def get_places(page):
    dictionary_places = {}

    for place in page['kind_of_places']:
        dictionary_places[place['id']] = place['name']

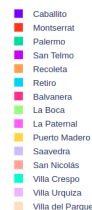
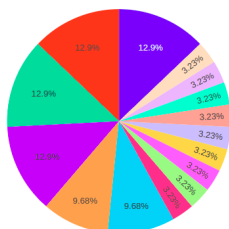
    return dictionary_places
```

```
1 import pandas as pd
2 import plotly.express as px
3 from dash import Dash, dcc, html, Input, Output
4 from jupyter_dash import JupyterDash
5 import pathlib
6 from datetime import date
7
8
9 path = str(pathlib.Path(__file__).parent.absolute())
10 hoy = date.today().strftime("%d-%m-%Y") # dd/mm/YY
11 file = path + f'/actividades_{hoy}.csv'
12
13 try:
14     df = pd.read_csv(file, sep='\t')
15 except:
16     # correr el scrapper? avisar que no corrió todavia?
17     df = pd.read_csv("/home/developer/training/proyecto_integrador_humai/data/prueba.csv", sep='\t')
18
19
20 df['count'] = df[df.columns[0]].count()
21
22 dftodas = df.groupby("activity_district").agg({"activity_id": "count"}).reset_index()
23
```

Porcentaje de actividades diario por localidad en la Ciudad de Buenos Aires



Cantidad de actividades por zona



Tipos de lugar:

Todos

Patrimonio

Taller de Oficios

Fachadas

Mercados y Ferias

Transporte

Todos

<>

Repositorio en Github del proyecto

https://github.com/Charz-a/proyecto_integrador_humai

Consideraciones finales

La experiencia fue muy satisfactoria. Nos encontramos con un desafío interesante que afrontar como equipo que nos aportó nuevos saberes no solo en lo técnico sino también en habilidades de organización y de comunicación.

Nos proponemos a futuro agregar una funcionalidad tal como comparaciones históricas de datos entre distintos días y una alerta en telegram de nuevas actividades publicadas en la web escapeada.

Agradecemos enormemente la asistencia del instituto humai en nuestro paso por los cursos y la realización del proyecto.