DSC 200 – Data Wrangling

Lab 6: Web Scraping

Goals:

- Identify and evaluate websites for scrapability.
- Use Python libraries such as pandas, BeautifulSoup, Selenium, and requests to extract and store data from a website.

Assignment Instructions:

Task 1: Evaluating Website Scrapability

Choose two websites that provide content you are interested in scraping. Verify their scrapability by reviewing their Terms of Use and robots.txt file.
For each website:

- Include an introduction (e.g., what the website is about, its domain, and why you selected it).
- Summarize the key web scraping policies outlined in the website's documentation.
- Attach the content of the robots.txt file as an appendix.
- Include URLs linking to their terms of service or any applicable scraping policies.

Deliverable:
Submit a well-structured two-page report in a Word document. Append the robots.txt contents for each website.

File Naming: group_[your_group_number]_Lab6_part1.docx

Task 2: Extracting Data from a Single Website

1. Choose one of the websites you evaluated in Task 1.
2. Write a Python function that:
- Retrieves content from a webpage allowed for scraping.
- Saves the content in a properly formatted CSV file.

Example Workflow:
- Use the requests library to fetch the webpage.
- Parse the content using BeautifulSoup.
- Extract data and save it in a CSV format.

Output File Format: group_[your_group_number]_task2.csv

Task 3: Scraping Specific Websites

1. Create a function to scrape data from the following website. Note that this includes pagination:
   - Data.gov (https://catalog.data.gov/dataset?q=&sort=views_recent+desc)
     - Extract the first 5 pages of data into a CSV file. The extracted data should include the following: datasetname, the source, description, csv_link, rdf_link, json_link, xml_link, zip_link, html_link, view_count
2. The function must:
   - Use pandas, BeautifulSoup, and requests libraries.
   - Save results to separate CSV files.
   - Print the number of rows and columns in each dataset.

   Output File Format:
   Name the files using this format:
   - group_[your_group_number]_task3.csv

Optional Challenge (Advanced):

Create a Python class that:
- Accepts a URL and output file name as parameters.
- Contains a method to retrieve and save webpage content.
- Use this class for all tasks.

Submission Details:

1. Part 1 Submission:
  - A Word document containing your website evaluation and robots.txt content.

2. Parts 2 & 3 Submission:
  - A Python script containing your web scraping functions.

File Naming Convention:
- Word Document: group_[your_group_number]_Lab_task1.docx
- Python Script: group_[your_group_number]_Lab_6.py

Evaluation Criteria:

- Clarity and structure of Task 1 report (20 marks).
- Proper implementation of the scraping function in Task 2 (20 marks).
- Accurate data extraction and saving for Task 3 (40 marks).
- (Optional) Advanced implementation using classes (10 bonus marks).

Notes: Always review the website's Terms of Use and robots.txt file before scraping. Ensure compliance with their policies and avoid scraping data that you are not authorized to access.