The first solution is selenium Webdriver.

Selenium webdriver is an open source tool used for automating web browsers. You can use Selenium Webdriver to automate the process of loading two webpages and comparing their elements, such as text images, and links.

Scenario 1 So in our first scenario we want to compare the html content of two webpages. I will showcase this by using my client code with the selenium web driver framework.

This is the client code in python:

from selenium import webdriver

# Create Chrome WebDriver instance

driver = webdriver.Chrome()

# Load the first webpage

driver.get("https://example.com")

# Get the HTML content of the first webpage

page1\_html = driver.page\_source

# Load the second webpage

driver.get("https://example.org")

# Get the HTML content of the second webpage

page2\_html = driver.page\_source

# Compare the two webpages by checking if the HTML contents are the same

if page1\_html == page2\_html: print("The webpages are identical")

else: print("The webpages are different") # Close the browser driver.quit()

**BeautifulSoup:**

* BeautifulSoup supports encoding detection which is a valuable feature that can yield better outputs for authentic HTML sites that do not fully disclose their encoding.
* Beautiful Soup is built on well-known Python parsers like lxml and html5lib, enabling us to experiment with various parsing techniques or trade off speed for flexibility.

The scenario here is simply for getting as much data out of a site as possible or potentially checking if the websites have a similar structure(Plagiarize for example )

Code that finds all html elements that use the “<p>” tag



Code that finds html elements that use the <p> tag but here selects the second element using the tag



**PyQuery**

PyQuery is a lightweight and fast python webscraper library. It bares similarity to the JQuery (for JavaScript). As its origin is comes from the JavaScript library, its functions and syntax also bares alot of similarity. It can parse HTML and XML documents and supports basic CSS selectors, which are

Unviersal selector

Selects all elements, but can also be restricted to a specific namespace.

Type selector

Selects all elements of a given node name

Class selector

Selects all elements of a given class

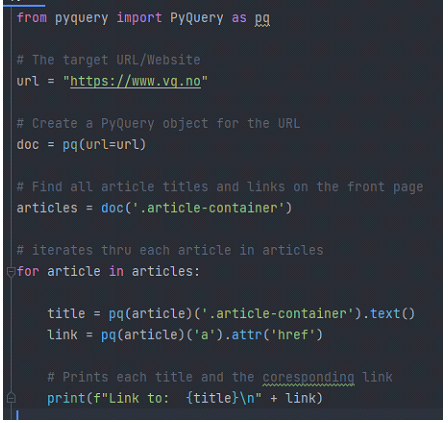
ID selector

Selects a specific element based on its ID

Attribute selector

Selects all elements with a given attribute

Client code:



Output (only an example as this code will output the whole page):



PyQuery is best suited for small-medium sized webscraping projects. The reason behind this is that pyQuery is built upon lxml which, as mentioned is made for fast parsing thru xml and HTML documents. It simply doesnt scale as well as a heavyer weigthed web scraper like for eksempel Scrapy.