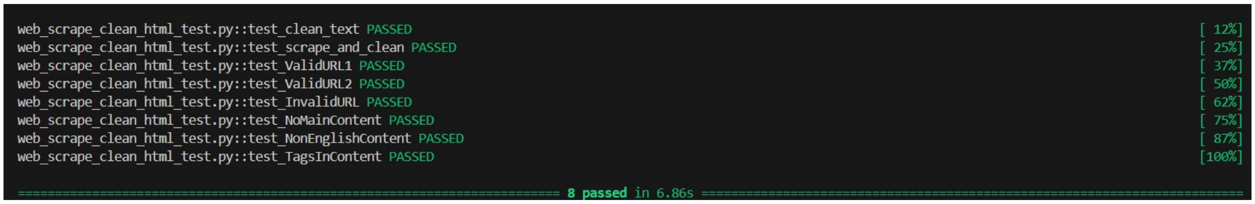
COSC 4P02 – HTML Web Scraper Test Document:

Completed Automated Tests:



```
web_scrape_clean_html_test.py::test_clean_text PASSED                                                              [ 12%]
web_scrape_clean_html_test.py::test_scrape_and_clean PASSED                                                        [ 25%]
web_scrape_clean_html_test.py::test_ValidURL1 PASSED                                                               [ 37%]
web_scrape_clean_html_test.py::test_ValidURL2 PASSED                                                               [ 50%]
web_scrape_clean_html_test.py::test_InvalidURL PASSED                                                              [ 62%]
web_scrape_clean_html_test.py::test_NoMainContent PASSED                                                           [ 75%]
web_scrape_clean_html_test.py::test_NonEnglishContent PASSED                                                       [ 87%]
web_scrape_clean_html_test.py::test_TagsInContent PASSED                                                           [100%]

======================================================================== 8 passed in 6.86s ========================================================================
```

Test Execution Commands:

python -m pytest web_scrape_clean_html_test.py -v

python -m pytest web_scrape_clean_html_test.py -s -v (this will display the messages and content printed in the tests when run)

python -m pytest web_scrape_clean_html_test.py -k "test_clean_text" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_scrape_and_clean" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_ValidURL1" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_ValidURL2" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_InvalidURL" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_NoMainContent" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_NonEnglishContent" -s -v

python -m pytest web_scrape_clean_html_test.py -k "test_TagsInContent" -s -v

Method Test Cases and Descriptions:

test_clean_text:

Test Case 1: Testing the “clean_text” method of the web scraper independently. This will confirm base functionality of the “clean_text” method. The method will replace newline and carriage return characters with spaces, replace quotes with pipe characters, and reduce multiple spaces to a single space.

Expected Result: Pass. The cleaned text should match the expected output.

test_scrape_and_clean:

Test Case 2: Testing the “scrape_and_clean” method with a valid URL independently. For this test, not a Wikipedia page, but a CNN page. This method will test the “scrape_and_clean” method on a valid URL. The method should scrape the page, clean the content, and return the page content.

Expected Result: Pass. The content scraped from the URL is a valid string with only permitted characters.

test_ValidURL1:

Test Case 3: Testing the “get_clean_content” method with a valid URL. For this test, Wikipedia. This method will test the “get_clean_content” method of the web scraper with a valid Wikipedia URL. The URL will be scraped, and the content will be cleaned. The content will be checked to ensure that it is a string and that it is not empty.

Expected Result: Pass. The content scraped from the URL is a valid string and is significantly long.

test_ValidURL2:

Test Case 4: Testing the “get_clean_content” method with a valid URL. For this test, CBS Sports. This method will test the “get_clean_content” method of the web scraper with a valid CBS Sports URL. The URL will be scraped, and the content will be cleaned. The content will be checked to ensure that it is a string and that it is not empty.

Expected Result: Pass. The content scraped from the URL is a valid string and is significantly long.

test_InvalidURL:

Test Case 5: Testing the “get_clean_content” method with an invalid URL. For this test, www.cosc4p02group9fakeurl.com.This method will test the “get_clean_content” method of the web scraper with a custom invalid URL. The web scraper returns "Error:" if the URL is invalid, so the test will check for this string in the content. If the string is found, the test will pass, the URL was invalid.

Expected Result: Pass. "Error:"" will exist in the content because the URL is invalid.

test_NoMainContent:

Test Case 6: Testing the “get_clean_content” method with a valid URL but no main content. For this test, https://www.example.com/.This method will test the “get_clean_content” method on a valid URL that doesn't have any main content. The page may be blank, or there may be no main content. The web scraper returns "Main content not found." if the main content is not found, so the test will check for this string in the content. If the string is found, the test will pass, the URL had no main content.

Expected Result: Pass. "Main content not found." will exist in the content because the URL has no main content.

test_NonEnglishContent:

Test Case 7: Testing the “get_clean_content” method with a valid URL that has content not in English. This method will test the “get_clean_content” method on a page with content in a language other than English. The web scraper should not include non-English lines and return only English text if present.

Expected Result: Pass. The content should not contain any lines that contain other languages.

test_TagsInContent:

Test Case 8: Testing the “get_clean_content” method with a valid URL that contains HTML tags embedded in the content. Citations, Footnotes, etc. This method will test the “get_clean_content” method on a page with HTML tags within the content. The web scraper should remove the HTML tags and return only readable text. Wikipedia articles contain many HTML tags, so this test could be performed on other Wikipedia articles to showcase the removal of embedded tags. Other sites also have tags, but Wikipedia is a good example.

Expected Result: Pass. The content should only contain clean text.