Summary Report on Selenium Testing with Python

Written by:

Matthew Brian Tahir

7 August 2023

[briantahirmatthew@gmail.com](mailto:briantahirmatthew@gmail.com)

LinkedIn: <https://www.linkedin.com/in/matthew-brian-tahir-725486251/>

# Introduction

This Selenium Testing Report documents the testing efforts conducted for the digital service website during my internship at BNI. The primary objective of this testing was to ensure the accessibility, usability, and performance of the application under test.

# Analysis of Areas for Improvements

As an intern during the testing process, I carefully analyzed the various test scenarios from BNI’s Test Plan Portal Digital Services and identified specific areas that could benefit from improvements. These areas include test case coverage, test data management, test environment setup, and test reporting.

In terms of test case coverage, I found that there’s quite many critical scenarios have been covered manually and a few for automation.

However, expanding the test case coverage to include edge cases and less common user scenarios is essential.

By doing so, we can enhance the overall robustness of the application and ensure that it performs well under various conditions.

Performance Testing:

In the performance testing phase, I assessed how well the digital service website performs under various load conditions.

By simulating a number of concurrent users and transactions, I evaluated its responsiveness, stability, and resource usage.

This testing helped identify potential bottlenecks and areas for optimization to ensure the application can handle real-world usage without any performance degradation.

Therefore, it has been concluded that the digital service website seemed to have loaded very well as it only took the page less than 5 seconds to receive the data.

Usability Testing:

Usability testing was an integral part of my evaluation process. I conducted tests to gauge how user-friendly the application is for end-users. By observing real users interacting with the application, I gathered valuable feedback on its intuitiveness, navigation, and overall user experience. The insights gained from usability testing were used to make user interface improvements, enhancing the overall usability of the application.

The provided code utilizes Selenium to conduct a user flow test for the website "<https://digitalservices.dglapm.id/register>," specifically testing the sign-up process. The script simulates a user signing up by entering relevant information such as email, first name, username, and password. It then submits the form and verifies if the sign-up process is successful by checking if the word "Welcome" is present in the page source. The code also captures a screenshot of the web page after the test. However, it's important to note that this code does not handle CAPTCHA or other interactive elements that require human intervention. Overall, this script aims to assess the usability of the website's sign-up process by automating the user flow test.

Accessibility Testing:

Lastly, Accessibility Testing is conducted to complement users on whether the language can exactly match the target Language of the original element being written.

Testing is done by checking that a specific element's text on a webpage matches the expected value.

This verification process can be relevant to accessibility testing because it ensures that critical information displayed on the webpage, such as labels or instructions, is accessible to users who rely on assistive technologies like screen readers.

In an accessibility context, it's also essential to validate that the content and elements on the website are perceivable and understandable by all users, including those with disabilities. Verifying the accuracy of the text in critical elements is part of that validation process.

Regarding test data management, I noticed that there is room for improvement to ensure data integrity and consistency across different test runs. Implementing data-driven testing techniques could increase the efficiency and effectiveness of our test cases, enabling us to test different data variations more thoroughly.

The test environment setup is a crucial aspect of obtaining reliable test results. During my analysis, I identified the need to focus on streamlining the test environment setup process and minimizing external dependencies. This will help reduce the possibility of encountering false positives or negatives during testing, ensuring a more accurate assessment of the application's performance.

In terms of test reporting, while the current structure of my testing report is comprehensive, I believe we can enhance it further by incorporating additional metrics. Metrics such as test execution time, defect density, and defect severity distribution can provide more insights into the application's quality and help us make data-driven decisions to improve our testing process.

# Limitations on selenium testing

By any means, Selenium is proven to be unsuitable for use in React-based applications because the xpath will change frequently due to the reliance on the testing techniques that could be used that were in line with the digital services framework

Most test scenarios (1-20) are well-suited for manual testing at the moment since the potential for automation testing lies in displaying uploaded images, as automation remains a powerful tool for automating repetitive tasks, accelerating testing processes, and enhancing overall quality, despite some instability in the testing results.

The automation faces difficulties in reading interactions with popups using locators due to the website's less responsive stability. The automation process still relies on manual interactions with elements directly.

Tasks like uploading images, copying image links, and adjusting image settings need to be completed manually. The complex XPath expressions make it challenging to inspect elements and may lead to overlapping elements in the code.

Automation is hindered by issues related to performance, fragility, readability, and maintenance, exemplified by complex XPath like //\*[@id='rc-menu-uuid-24321-2-miscellaneous-popup']/li[3].

# Strategies for implementing the testing

**In conclusion, the information about testing efforts for the Digital Services BNI website have provided valuable insights into its accessibility, usability, and performance.**

**The analysis highlighted areas for improvements, particularly in the automation and element locating process.**

**To overcome the challenges faced during Selenium testing, it is essential to adopt more straightforward, robust, and maintainable strategies for finding elements on a website, preferably through a react testing framework.**

This can be achieved by using stable and unique identifiers like IDs or classes whenever possible, which offer reliable element references.

Additionally, leveraging CSS selectors as element locators can provide a more efficient approach to target elements based on their attributes.

Emphasizing the use of semantic tags in the website's structure aids in better identifying and interacting with critical elements.

Furthermore, opting for relative XPATH expressions can improve code readability and maintainability, reducing the risk of overlapping elements.

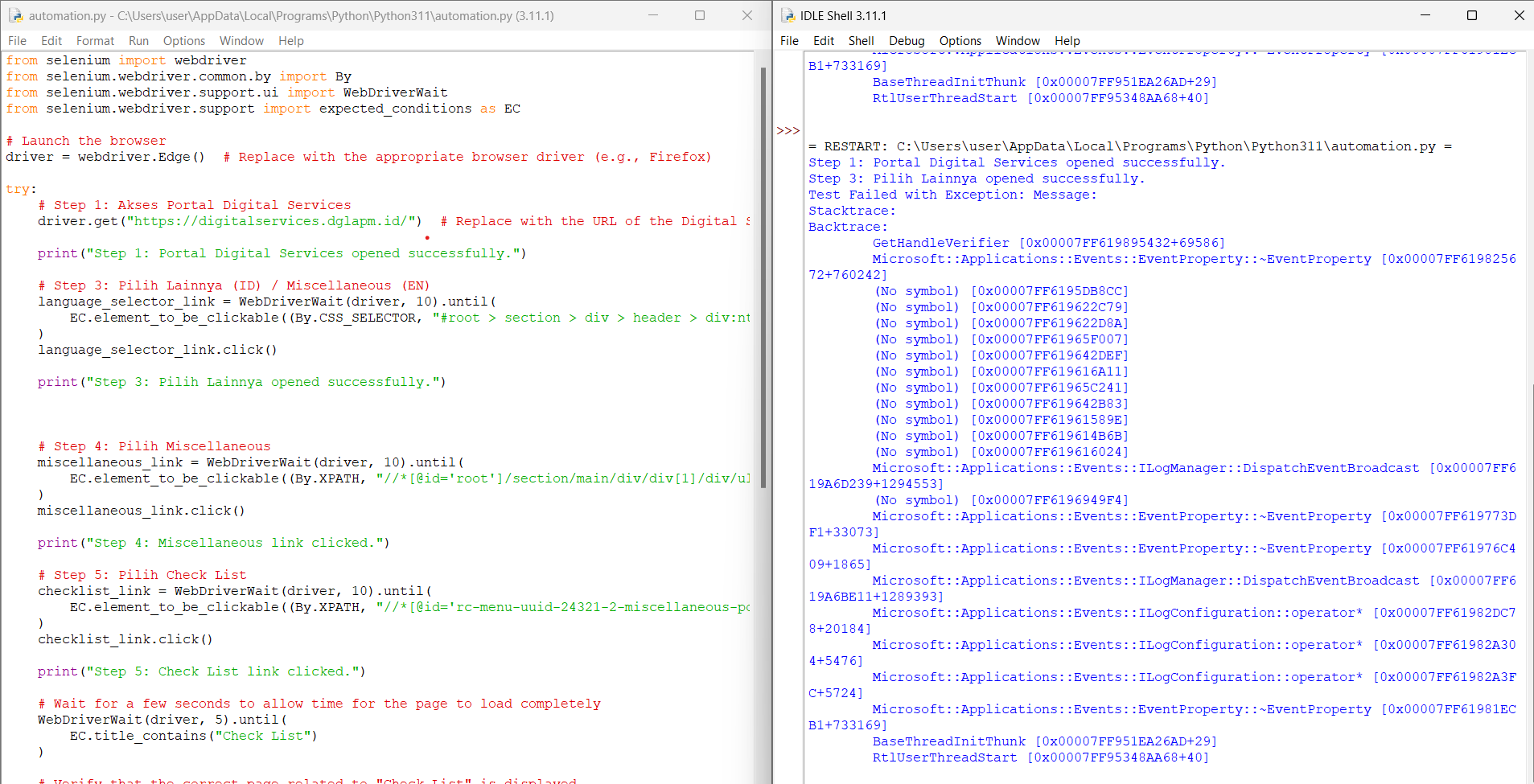
By implementing these strategies, the testing process can become more efficient and reliable, ensuring a better-performing and user-friendly Digital Services BNI website.

Overall, continuous testing enhancements and refinements will contribute to the application's success and adherence to the highest standards of quality, accessibility, and usability.

# Appendix

Code for testing:  
Akses Portal Digital Services

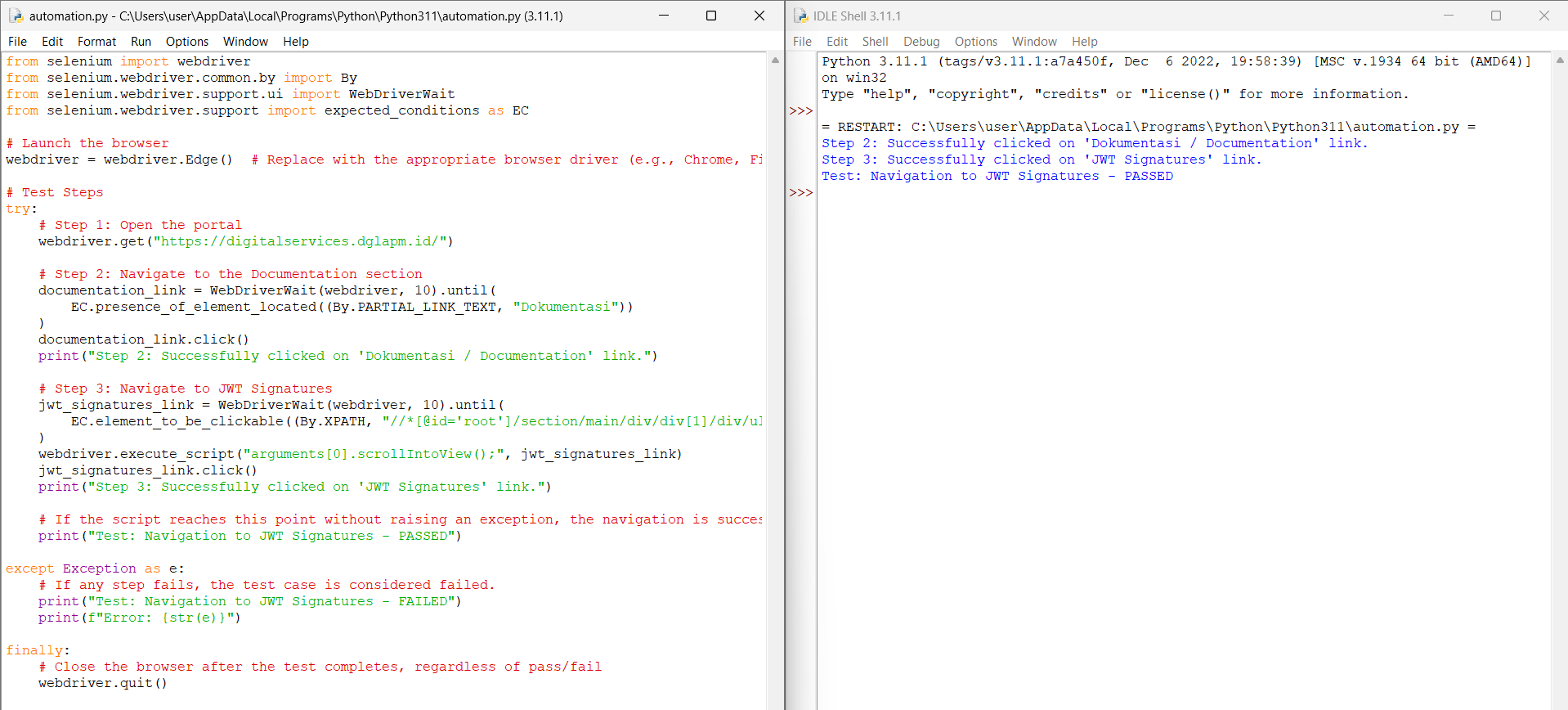
* Pilih Dokumentasi (ID) / Documentation (EN)
* Pilih Lainnya (ID) / Miscellanueous (EN)
* Pilih Miscellaneous
* Pilih Check List

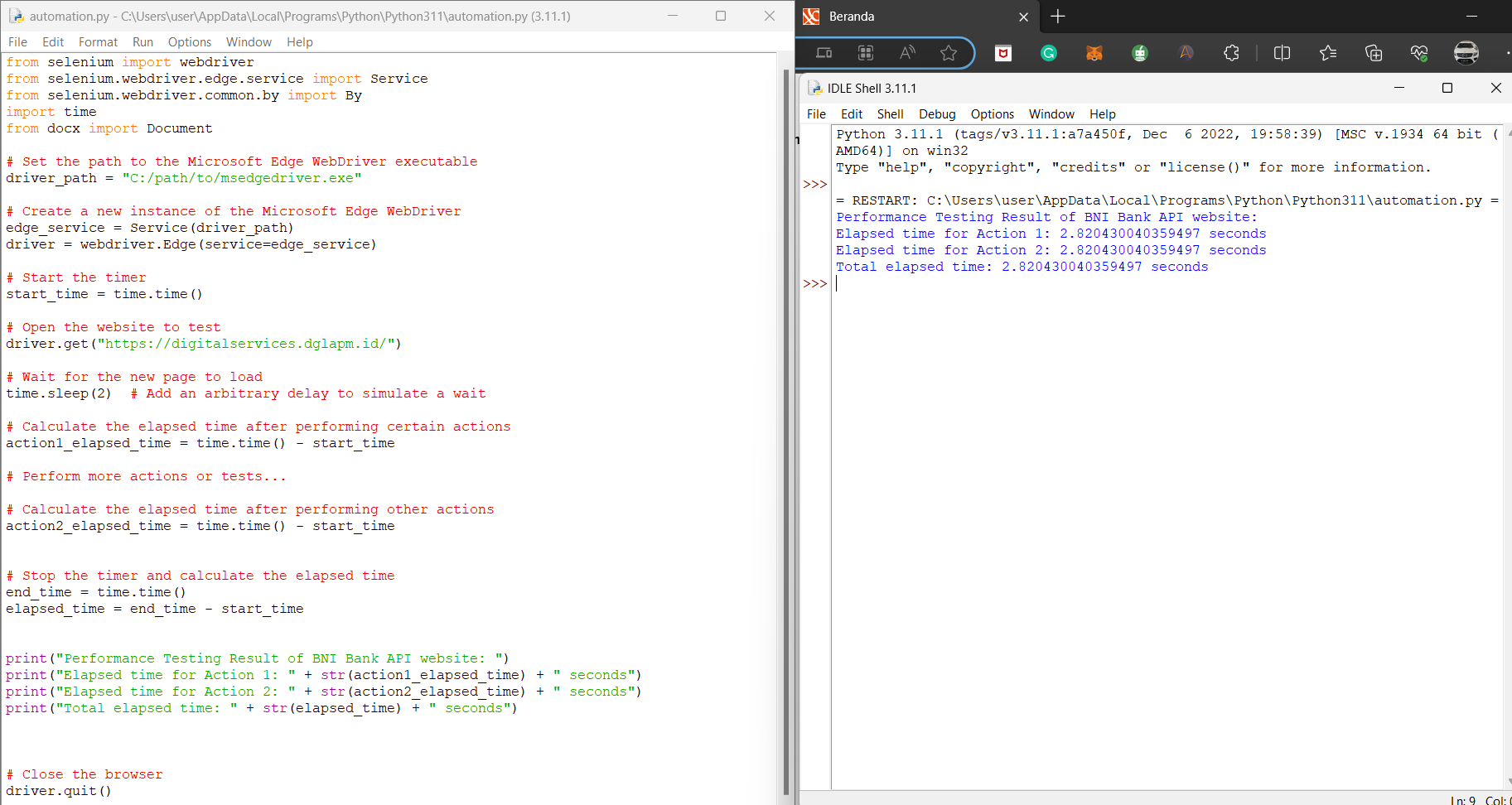


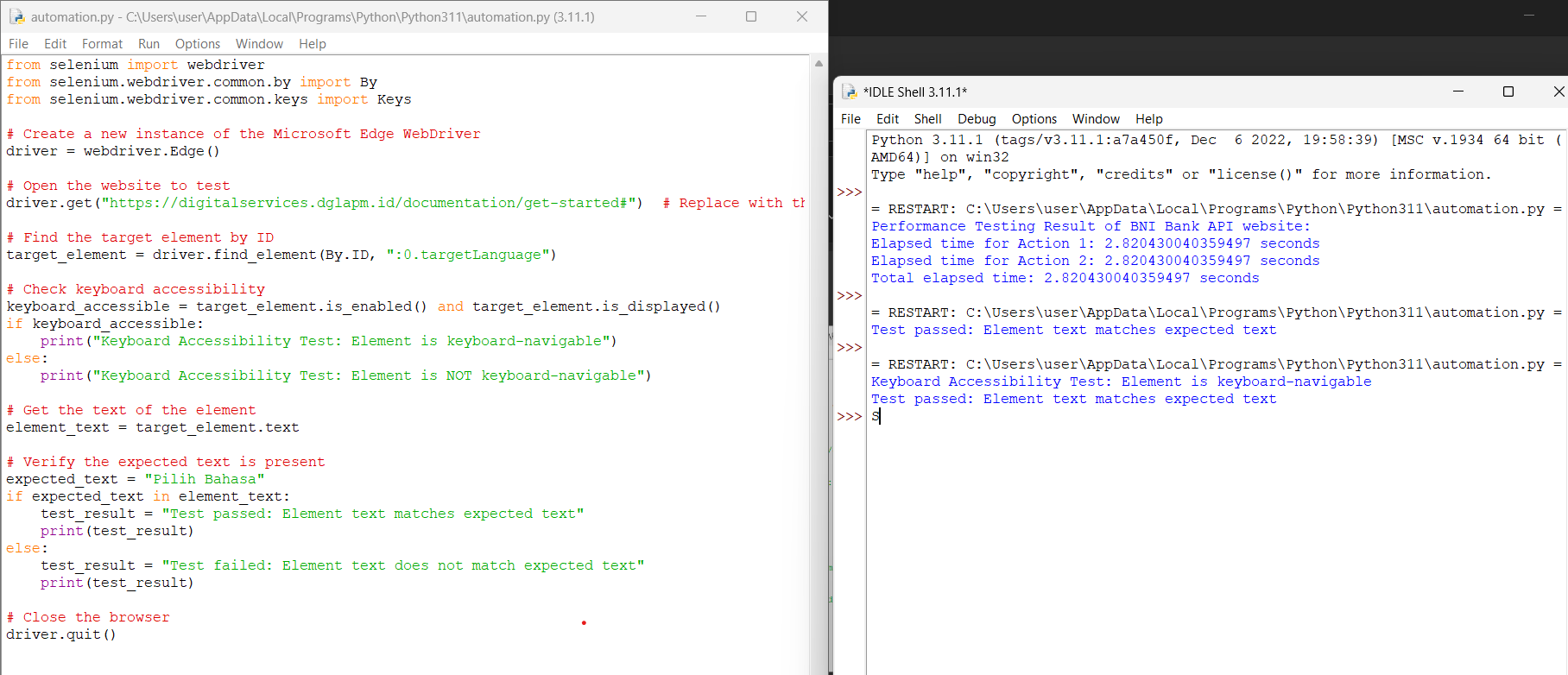
Code for testing the navigation of the website:  
Akses Portal Digital Services

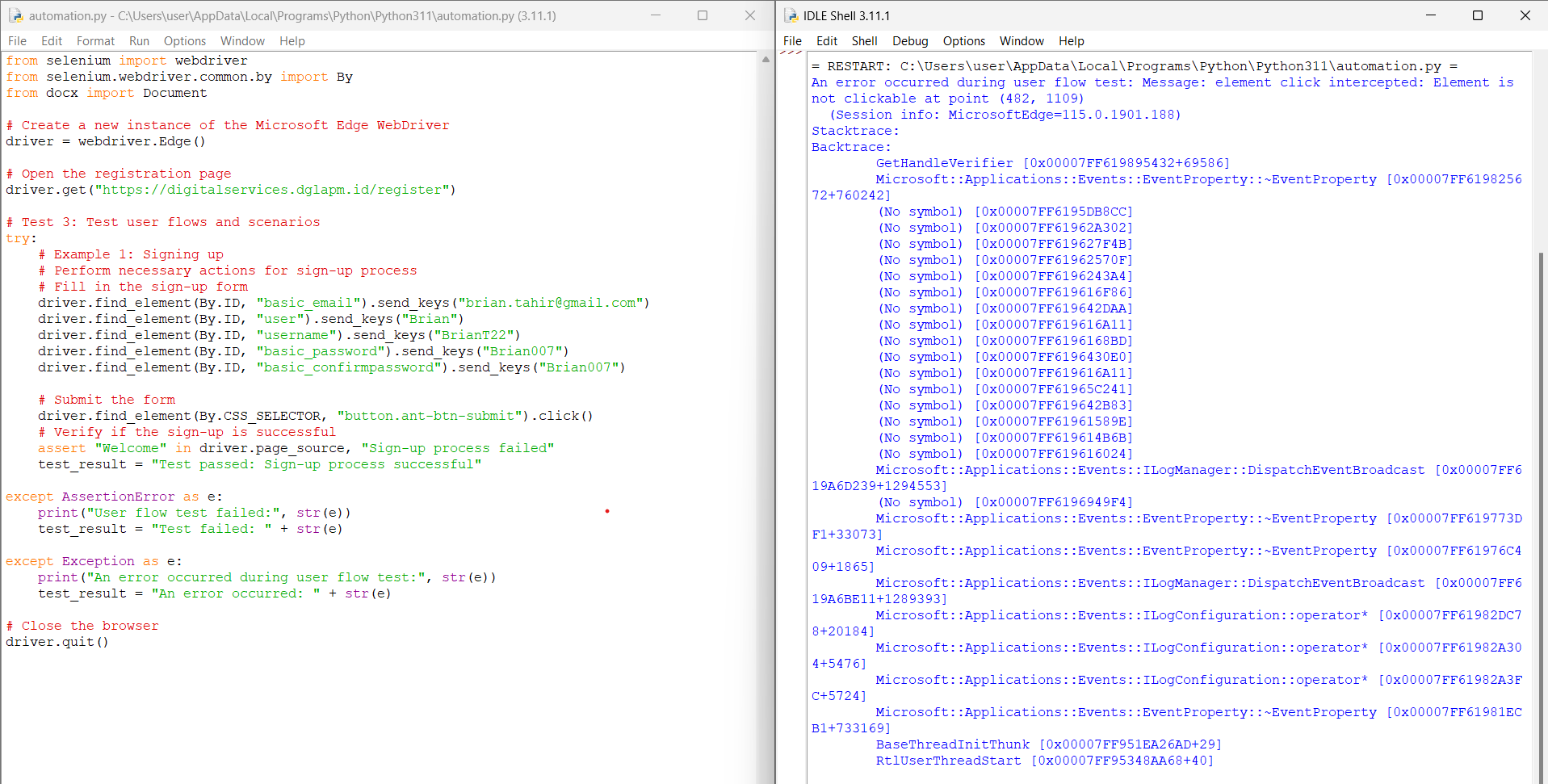
* Pilih Dokumentasi / Documentation
* Pilih JWT Signatures
* Melakukan pengecekan/ validasi content/informasi yang adai di JWT Signature Service
* Test Fungsi Copy & Show Snippet Code

\*Test Fungsi Copy & Show Snippet cukup perlu manual testing

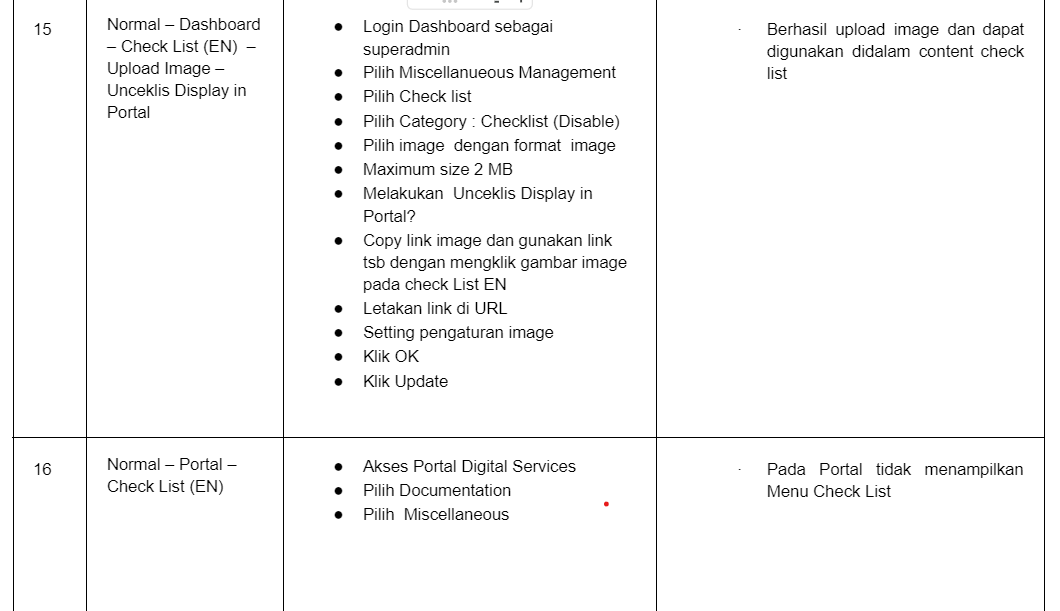


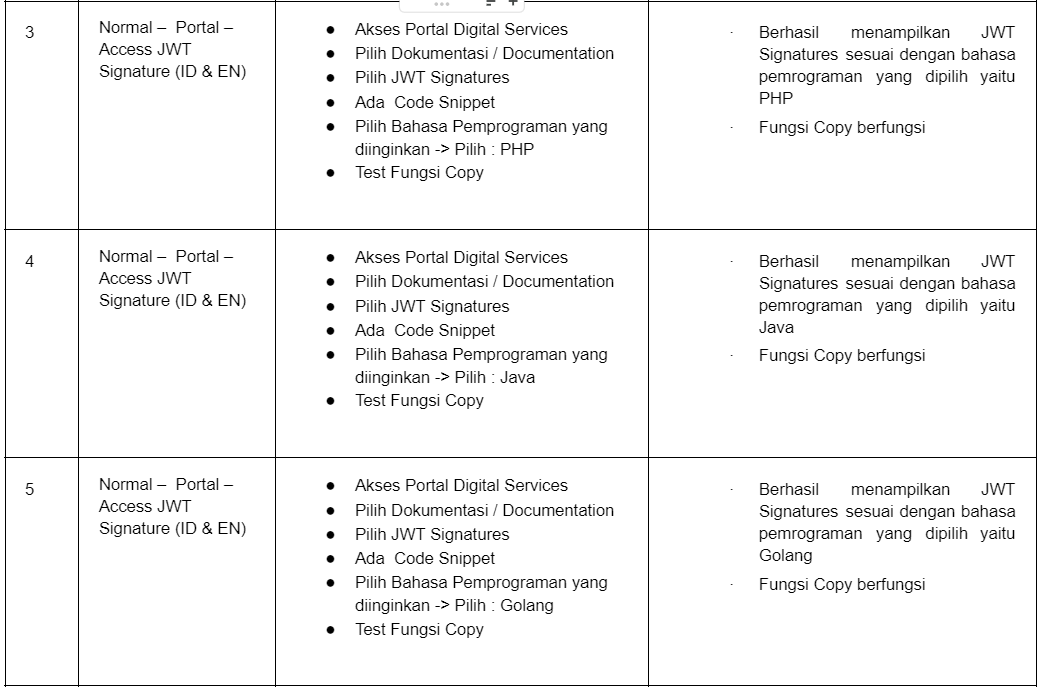
Code for performance testing:  


Code for accessibility testing:  


Code for usability testing:  


Example test scenarios where manual testing is effective:





Examples of absolute XPath:

* login\_button = webdriver.find\_element(By.XPATH, "//\*[@id='normal\_login']/div[3]/div/div/div/div/button")
* miscellaneous\_management\_link = WebDriverWait(webdriver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id='13']/div"))

)

* check\_list\_title\_element = WebDriverWait(webdriver, 10).until(

EC.presence\_of\_element\_located((By.XPATH, "//\*[@id='rc-menu-uuid-332-7-/miscellaneous-popup']/li[2]/span[2]/a"))

)

Rungta, K. (2023). *XPath in Selenium: How to Find & Write? (Text, Contains, AND)*. [online] www.guru99.com. Available at: ‌[XPath in Selenium: How to Find & Write? (Text, Contains, AND)](https://www.guru99.com/xpath-selenium.html) [Accessed 7 Aug. 2023].

*16 Selenium Best Practices For Efficient Test Automation*. [online] LambdaTest. Available at: <https://www.lambdatest.com/blog/selenium-best-practices-for-web-testing/>[Accessed 7 Aug. 2023].

SmartBear.com. (n.d.). *Quick Tips To Improve Your Selenium Testing With Python*. [online] Available at: <https://smartbear.com/blog/quick-tips-to-improve-your-selenium-testing-with-p/#:~:text=presence_of_element_located%20visibility_of%20invisibility_of_element_located>[Accessed 7 Aug. 2023].

‌

‌

‌