SDET - Coding Challenge

(Walk-in 16th December 2023, CAW Studios)

UI Automation Scenario: Swag Labs e-commerce

Problem Statement:

As part of our coding challenge, we want candidates to automate the ecommerce website. The objective is to ensure that users can seamlessly sort the products, store its details, and successfully add them to the shopping cart.

Requirements:

Automation Tool and Language:

- Candidates are free to choose any automation tool of their preference (e.g., Selenium, Cypress, Playwright) and any programming language (e.g., Java, Python, JavaScript).
- The chosen tool and language should align with best practices for UI automation.

Script Structure and Readability:

- Candidates should organize their automation scripts in a clear and modular structure.
- Code should be well-documented with comments to explain the purpose of each section and important steps.
- Follow coding standards and naming conventions for improved readability.

Reusable Components:

- Encourage candidates to create reusable functions or methods for common actions, such as navigating to a page, interacting with elements, and verifying outcomes.
- Prevent duplicating of test data

Efficient Locators and Waits:

- Ensure that candidates use efficient and robust locators to identify UI elements.
- Implement appropriate wait strategies to handle dynamic loading and ensure script stability.

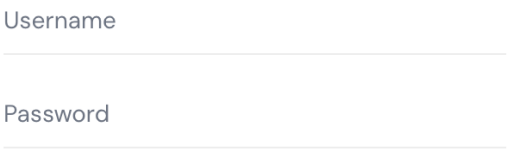
Assertions and Reporting:

- Include meaningful assertions to validate that the product sort and add-to-cart actions are successful.

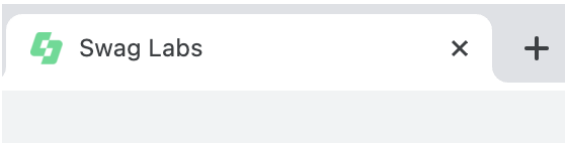
Testing scenario:

AUT: https://www.saucedemo.com/

1. Navigate to AUT
2. Validate whether username and password fields are editable

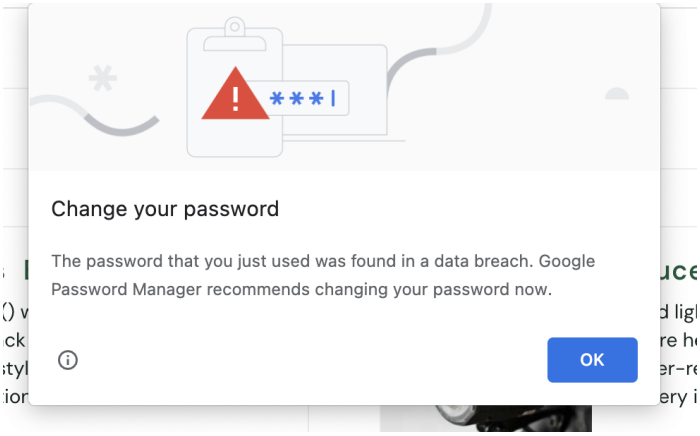


3. Validate the page title (Swag Labs)



4. Get the username and password details from the page and login

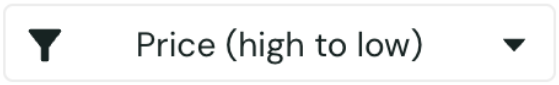
5. Handle the password expiry browser alert and proceed



6. Validate whether sort dropdown is displayed

7. Validate whether the products are displayed according to ascending order of Name by default

8. Change the sorting order of price (high to low)



9. Validate whether the products are displayed according to descending order of price
10. Read all the product names and prices and store it in a json file as an object with key and value pairs

```
{
  "productname1": {
    "price": "15.99",
    "description": "product1 description"
  },
  "productname2": {
    "price": "19.22",
    "description": "product2 description"
  }
}
```

11. Add all the products to the cart
12. Validate the number of items displayed on cart icon

13. Click on cart icon to navigate to checkout page
14. Validate all the item names, prices and descriptions that are displayed in the cart are valid (the data should be read from the json file that was created earlier)
15. Click on checkout button
16. Validate firstname, last name and Zip code fields are editable
17. Provide the details and click on continue button
18. Validate whether the total price is correct (sum of all prices of added items + fixed tax)

Price Total
Item total: $129.94
Tax: $10.40

Total: $140.34

19. Click on finish button and validate the thank you message

Thank you for your order!

20. Click on the sandwich menu on top left and logout, validate user is logged out.