README:

* Framework used – Python, Selenium and Pytest Framework
* Prerequisites prior to running test-

1. Check python is installed in cmd using:

C:\Users\Ankita>python --version

Python 3.11.4

If not install python in your system and add the path to your system environment variables

1. Install selenium for python using:

C:\Users\Ankita>pip install selenium

C:\Users\Ankita>pip show selenium

Name: selenium

Version: 4.19.0

1. Chrome driver version: 123

Download link: <https://storage.googleapis.com/chrome-for-testing-public/123.0.6312.122/win64/chromedriver-win64.zip>

(used chrome, but code can be changed to incorporate different browser such as firebox, Edge etc.)

1. Install pytest : C:\Users\Ankita>pip install pytest

C:\Users\Ankita>pytest --version

pytest 8.1.1

1. Install pytest-html: C:\Users\Ankita>pip install pytest-html

* Once above packages are installed

Run pytest using: py.test --html=report.html

* Code in GitHub Path: <https://github.com/adey512/pythonTesting>. For the below test cases please refer to test file test\_T001.py, test\_T002.py, test\_T003.py, test\_T004.py, test\_T005.py, test\_T006.py, test\_T007.py and there are sub test cases under the files as mentioned in the document
* View reports in report.html file
* Test Cases Covered:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Case Name | Test Case Description | Expected Result | Actual Result |
| Standard User | | | | |
| T001 | TestLogin  Positive Test | 1. Verify login url page | Correct url <https://www.saucedemo.com/> |  |
| 1. Log in with a valid Username (standard\_user) and Password (secret\_sauce) and user is redirected to inventory page url | Inventory page is successfully displayed, and URL is <https://www.saucedemo.com/inventory.html> |  |
| T002 | Login page validation – Negative test | 1. Enter an incorrect password for a valid username | Error Message: Username and password do not match any user in this service |  |
| 1. Enter an incorrect username for a valid password |  |
| 1. Login with a valid username and case-insensitive password. |  |
| 1. Enter an empty password field | Error Message: Password is required |  |
| 1. Enter an empty username field | Error Message: Username is required |  |
| T003 | Products inventory tests | 1. Check if the products list is not empty. (This can be modified to check the count of total items in the inventory) | Assert inventory list >0 |  |
| 1. Click on Sort by Price (low to high) | Get before sort inventory price (without sort) Sort using code.  Select dropdown low to high (browser sorting). Check if the two lists are equal |  |
| 1. Click on Sort by Price (high to low) | Get before sort inventory price (without sort) Sort using code in desc order of price.  Select dropdown low to high (browser sorting). Check if the two lists are equal |  |
| 1. Click on Sort by Name (A to Z) | Assert inventory list is sorted by Name (A to Z) |  |
| 1. Click on Sort by Name (Z to A) | Assert inventory list is sorted by Name (A to Z) |  |
| 1. Test to check if clicked on an inventory name, it is redirected to point to correct inventory item page | The clicked inventory page should match the product name in the inventory detail page |  |
| T004 | Test to add and remove products to cart and checkout | 1. Add any no of products to cart (use randomize function) and click on cart | Check the count. Assert the number of items is same as added |  |
| 1. Validate add to Cart Page url | The correct url should be |  |
| 1. Count number of items in the cart | Check the count. Assert the number of items is same as added in previous step |  |
| 1. Click on remove button to remove product | Check the count. Assert the number of items is reduced |  |
| 1. Click on continue Shopping button. Check if back to inventory url page | Assert the page url is the inventory list. |  |
| 1. Click back to add to cart Page and recheck if items are added to cart | Check the count. Assert the number of items |  |
| 1. Click on Checkout. Validate if redirected to checkout-step 1 page | Assert if redirected to checkout- step-one-page url |  |
| 1. Cancel from Checkout step one page. Validate if redirected to add to cart page | Assert if redirected to add to cart URL |  |
| 1. Click continue and back to checkout page | Assert page url and cart count |  |
| 1. Cancel from Checkout step two page. Validate if redirected to add to inventory page | Assert if Inventory page is successfully displayed |  |
| 1. Take prices of items from the cart and sum them up. Check Item sun-total price (before tax) | Assert if summation of prices from cart inventory is equal to the sun total price displayed |  |
| 1. Cancel from Checkout step two page. Validate if redirected to add to inventory page | Assert if redirected to add to inventory page |  |
| 1. Checkout page with empty first name | Error message First Name is required is displayed |  |
| 1. Checkout page with empty last name | Error message Last Name is required is displayed |  |
| 1. Checkout page with empty postal code | Error message postal code is required is displayed |  |
| 1. Enter first name, last name, zipcode and click Continue | Assert if redirected to checkout-step-two page url |  |
| 1. Finish from Checkout step two page. Validate if redirected to check out complete page | Assertif redirected to check out complete page |  |
| 1. Click on Back Home | Assert if the url is redirected to products page and number of items added in cart is 0 |  |
|  |  | 1. Try to access cart or inventory pages without logging in | Error message: Cannot access without login |  |
| T005 | Logout | 1. Click on logout button from side Menu | Assert if back to the main url <https://www.saucedemo.com/> |  |
| Locked out user | | | | |
| T006 |  | 1. Log in with a valid Username and Password for locked out user | Error Message : Sorry, this user has been locked out |  |
| Problem User | | | | |
| T007 |  | 1. Log in with a valid Username and Password for problem user | Check if login is successfully and page is redirected to inventory page |  |
|  | 1. Validate if image is broken in inventory Page | Get img using get attribute and raise exception when broken/missing/invalid schema is encountered |  |
|  | 1. Validate checkout user information page | Raise exception when cannot continue to checkout step two page. Return to login page |  |

* Scalability:

The proposed solution works for small to medium website traffic/load.

The reasons for choosing the selenium -Python framework with Pytest involves-

1. Modular framework that divides the test cases into small and independent modules.
2. We can create a layer of abstraction between the test scripts and the application under test. This makes our test scripts more readable, maintainable, and reusable, as this only needs to update the page objects when there are changes in the AUT, rather than modifying each test script individually. In my case I have used conftest.py for storing urls, login details etc.
3. This framework can be integrated easily with Continuous integration (CI) and continuous delivery (CD) practices that enable us to integrate our code changes frequently and deliver software products faster and more reliably.

For more content and higher traffic, we need to have a way to parallelize the tests. To make it more scalable we may need to consider implementing parallel testing as follows:  
Parallel testing is a technique that allows us to run multiple test scripts or test cases simultaneously on different browsers, devices, or environments. This can significantly reduce the test execution time, improve test coverage, and increase feedback speed. Parallel testing can also help us to scale the test automation framework design pattern, as we can easily add or remove test scripts or test cases without affecting the rest of our test suite. To implement parallel testing, we need to use a tool or a service that can distribute our test scripts or test cases across multiple machines or servers, such as Selenium Grid, TestNG, or a cloud-based platform.