Automated Testing with Selenium

About Selenium

Selenium is a popular open-source framework for automating web browsers. It offers a toolkit for online scraping and web application testing, making it a great resource for developers, testers, and quality assurance specialists. Selenium is particularly well-liked in the area of automated testing, where it is utilized to verify the functionality of web applications across various platforms and browsers.

About Trello

Trello is a project management and collaboration tool that uses a visual board and card system to help individuals and teams organize tasks, projects, and workflows.

Test Scenario Identification :

Key Features of Trello:-

- Boards:
  Trello boards are the central workspace for organizing projects and tasks visually. Each board represents a project or category and contains lists to categorize work into stages or phases. Lists are customizable, and cards within lists represent individual tasks with details, due dates, and checklists. Trello's intuitive drag-and-drop interface makes it easy to manage and track work on these boards.
- Lists and Cards:
  Lists in Trello are the vertical columns on boards that define the various stages or categories of work. They can be customized to match specific project workflows, providing an at-a-glance view of the task's progress.

  Cards, on the other hand, are individual task items within lists, serving as actionable units. They allow users to detail and prioritize specific tasks, complete with descriptions, due dates, attachments, and checklists. Cards are essential for organizing, collaborating, and tracking the progress of

individual tasks within a project, making Trello a versatile tool for project management and task tracking.

- Sharing and Collaborating:
  Trello supports real-time collaboration, enabling team members to comment on cards, attach files, mention colleagues, and keep conversations organized within specific tasks.

- Filtering and Searching:
  Trello supports real-time collaboration, enabling team members to comment on cards, attach files, mention colleagues, and keep conversations organized within specific tasks.

Test Scenarios and Functionalities to be tested by Selenium:

We have tested the following functionalities:-

1. Creating Boards:-
   Conducted a test of Trello's board management functionality with Selenium by creating new projects and verifying their successful creation.
2. Managing Lists
   Used Selenium to test Trello's list management capabilities by creating and manipulating lists within the application
3. Adding Cards and Lists
   Determined whether Trello seamlessly handled image attachments and various content formats, such as text, files, and multimedia.
4. Testing the login page
   Tested to verify that users can successfully create accounts and log in.

Other functionalities identified that can be tested:

1. Notifications
   - We can test the notification functionality, including alerts for updates to boards and cards.
   - Verify that users receive notifications for mentions and due dates.

2.Collaboration and Sharing:
   - Ensure that multiple users can collaborate on the same board.
   - Test permissions and access control for shared boards.

3. Mobile Responsiveness:
   - Test Trello's mobile responsiveness on various devices and screen sizes.

4. User Permissions and Roles:
   - Verify that different user roles (e.g., admin, member, etc) have the correct permissions on boards and cards.

5. Accessibility:
   - Check if Trello is accessible to users with disabilities and adheres to accessibility standards.

6. Integration with Other Tools:
   - If Trello integrates with other apps or services, test these integrations to ensure data flows correctly.

7. Search and Filter:
   - Test the search functionality to find boards, cards, and users.
   - Verify that filtering options work as expected.

2. Test Script Creation:

```python
@pytest.mark.usefixtures("setup")
class TestBoard:
    board_name = "test"
    list_name = "testList1"
    cardName = "card1"
    driver: WebDriver

    def test_create_board(self):

        navigate_to_boards_page(self.driver)

        self.driver.find_element(By.XPATH, value: "//div[@class='board-tile mod-add']").click()
        board_title = self.driver.find_element(By.XPATH, value: "//input[@type='text']")
        board_title.send_keys(self.board_name)
        sleep(2)
        board_title.send_keys(Keys.ENTER)
        # buttons = self.driver.find_elements(By.TAG_NAME, "button")
        #
        # for button in buttons:
        #     if button.text == "Create":
        #         button.click()
        #         break
        sleep(5)

        assert self.board_name in self.driver.title
```

```python
    def test_add_list(self):
        navigate_to_boards_page(self.driver)
        if not is_selected_board_displayed(self.driver, self.board_name):
            if not select_board(self.driver, self.board_name):
                assert False, "Board does not exist"
        try:
            add_list_button: WebElement = WebDriverWait(self.driver, timeout: 13).until(
                lambda x: x.find_element(By.XPATH, '//*[@id="board"]/div/button')
            )
            add_list_button.click()
            list_title = self.driver.find_element(
                By.XPATH,
                value: '//*[@id="board"]/div[1]/form/textarea')
            list_title.send_keys(self.list_name)
            list_title.send_keys(Keys.ENTER)
            list_elements = self.driver.find_elements(By.TAG_NAME, value: "h2")
            added = False
            for list_element in list_elements:
                if list_element.text == self.list_name:
                    added = True
            assert added, "Failed to add a list"
        except NoSuchElementException:
            assert False, "Failed to find a list"
```

4. Reporting:

report.html

Report generated on 24-Oct-2023 at 12:26:59 by pytest-html v4.0.2

Environment

Python	3.8.1
Platform	Windows-10-10.0.22000-SP0
Packages	• pytest: 7.4.2 • pluggy: 1.3.0
Plugins	• html: 4.0.2 • metadata: 3.0.0

Summary

4 tests took 00:01:27.

(Un)check the boxes to filter the results.

0 Failed, 4 Passed, 0 Skipped, 0 Expected failures, 0 Unexpected passes, 0 Errors, 0 Reruns — Show all details / Hide all details

Result	Test	Duration	Links
Passed	Tests/test_authentication.py::TestLogin::test_login	00:00:05	
Passed	Tests/test_board.py::TestBoard::test_create_board	00:00:12	
Passed	Tests/test_board.py::TestBoard::test_add_list	00:00:16	
Passed	Tests/test_board.py::TestBoard::test_add_card_to_list	00:00:08	

Implemented the reporting functionality using *"pytest-html"* which directly generates an HTML Report.

The report indicates that we passed all the test cases in the functionalities of :-

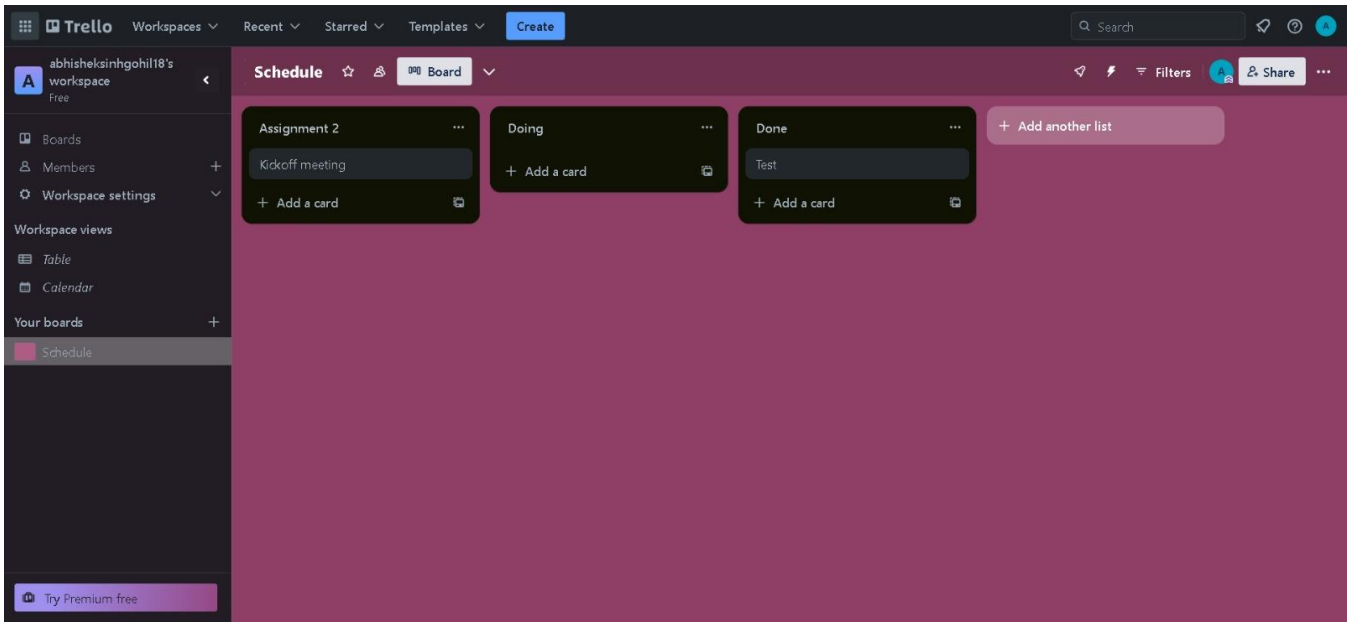
- Authentication
- Creating a board
- Adding Lists
- Adding Cards to List

Along with the results of test cases being passed/failed, The plugin also reports the amount of time taken to test the feature.

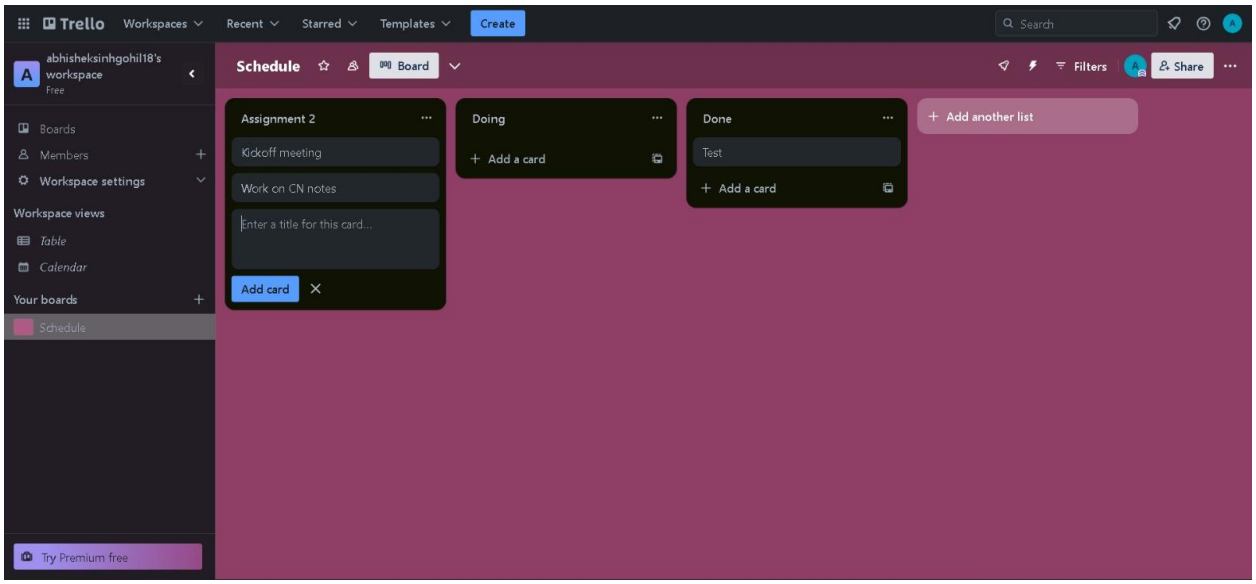
Test Execution and Documentation

After executing the tests using the script:

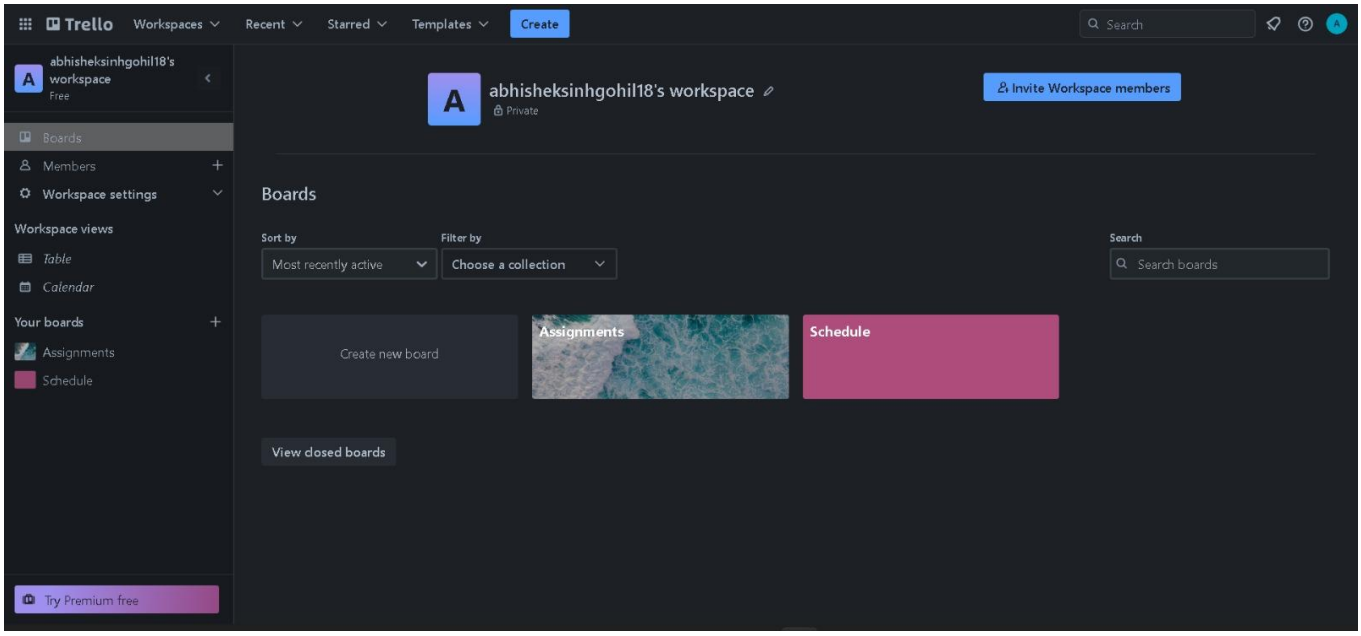
1.Lists have been added successfully



2. Card have been added successfully



3.Boards have been added successfully



4. Login