# Automated Testing with Selenium

## Selenium:

Selenium is an open-source tool implemented in the automation of web browsers. It works by enabling you to write scripts of interaction with web elements, emulating user actions, or running tests on web applications. Its major area of application is in testing and automating web applications

## Test Scenario Identification

1. Verify login functionality with both valid and invalid credentials
   Automate the login authentication into Trello Website

2. Verify the creation of new boards, lists, and cards.
    Automate creation of new board, new list and new card after login

3. Verify drag-and-drop functionality for moving cards between lists.
   Automate the movement of moving a card from one list to another

4. Verify that specific elements (like buttons, links, and input fields) are correctly displayed on the dashboard
   Automate navigation to main dashboard and identify and verify specific elements are displayed in the respective dashboard.

5. Verify proper functionality of filtering, labeling, and search features
   Automate testing functionality of filtering using labels, labelling using custom labels to different cards and search using keyword in main search bar.

## Test Script Creation

Software Used : VS Code

Language : Python

Browser: Chrome

Libraries: coverage, pytest, pytest-cov, pytest-html, pytest-metadata, pytest-dotenv,webdriven-manager

# Reporting

**report.html**

Report generated on 02-Nov-2024 at 23:11:09 by pytest-html v4.1.1

**Environment**

| Python | 3.13.0 |
|---|---|
| Platform | Windows-11-10.0.22631-SP0 |
| Packages | • pytest: 8.3.3 <br> • pluggy: 1.5.0 |
| Plugins | • cov: 6.0.0 <br> • html: 4.1.1 <br> • metadata: 3.1.1 |

**Summary**

9 tests took 00:02:08.

(Un)check the boxes to filter the results.

☑ 0 Failed, ☑ 9 Passed, ☑ 0 Skipped, ☑ 0 Expected failures, ☑ 0 Unexpected passes, ☑ 0 Errors, ☑ 0 Reruns

Show all details / Hide all details

| Result ▲ | Test | Duration | Links |
|---|---|---|---|
| Passed | test_check_elements.py::test_trello_dashboard | 00:00:11 | |
| Passed | test_create_board_and_list.py::test_trello_create_board_and_list | 00:00:19 | |
| Passed | test_create_card.py::test_trello_create_board_list_and_card | 00:00:24 | |
| Passed | test_drag.py::test_drag_and_drop_card | 00:00:12 | |
| Passed | test_filter.py::test_filter_by_green_label | 00:00:14 | |
| Passed | test_invalid_login.py::test_invalid_password_login | 00:00:09 | |
| Passed | test_labelling.py::test_add_green_label_to_card | 00:00:16 | |
| Passed | test_login_2.py::test_trello_login | 00:00:10 | |
| Passed | test_search.py::test_trello_login_and_search_navigation | 00:00:13 | |

I developed a report using pytest that consolidates and generates a HTML report of all test cases with coverage.

Above report indicates, all functional and non-functional requirements were included in the test scenarios and all the test cases are passed successfully.

# Test Execution and Documentation

**1.Verify login functionality with both valid and invalid credentials**
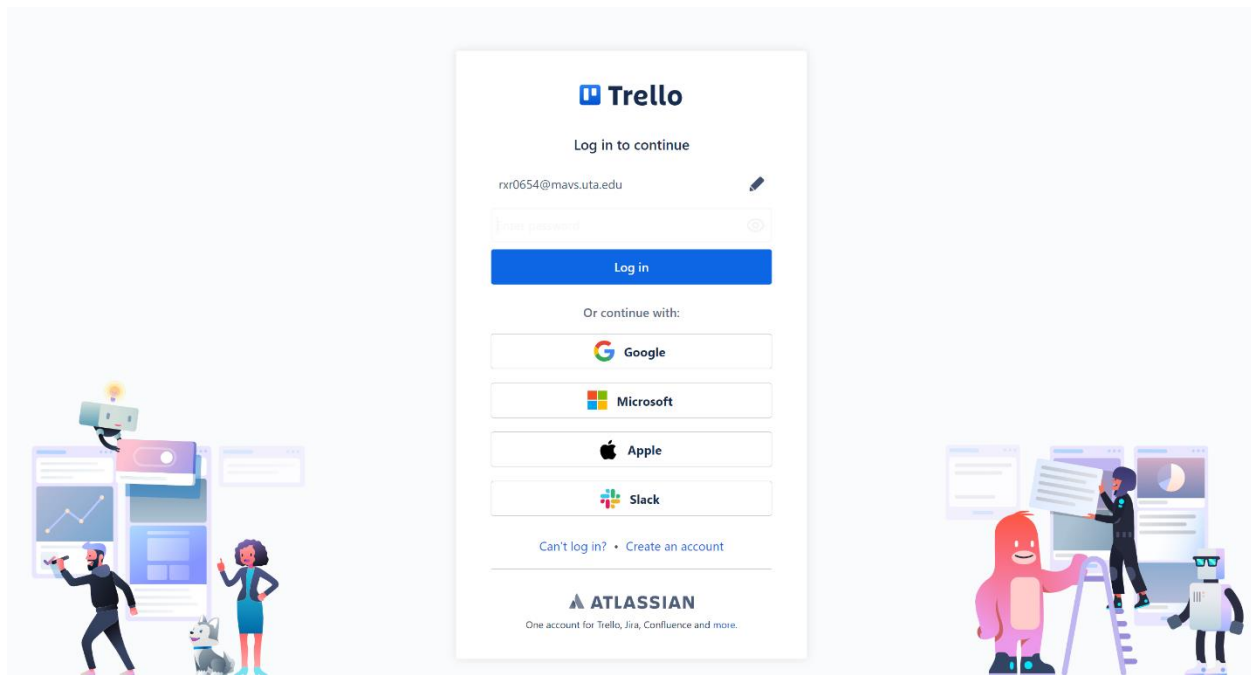
**Automate the login authentication into Trello Website**

Valid Credentials

**Test Data:**
**Email: rxr0654@mavs.uta.edu**
**Password: <: My password>**

Input Email



Input Password

Login Successful

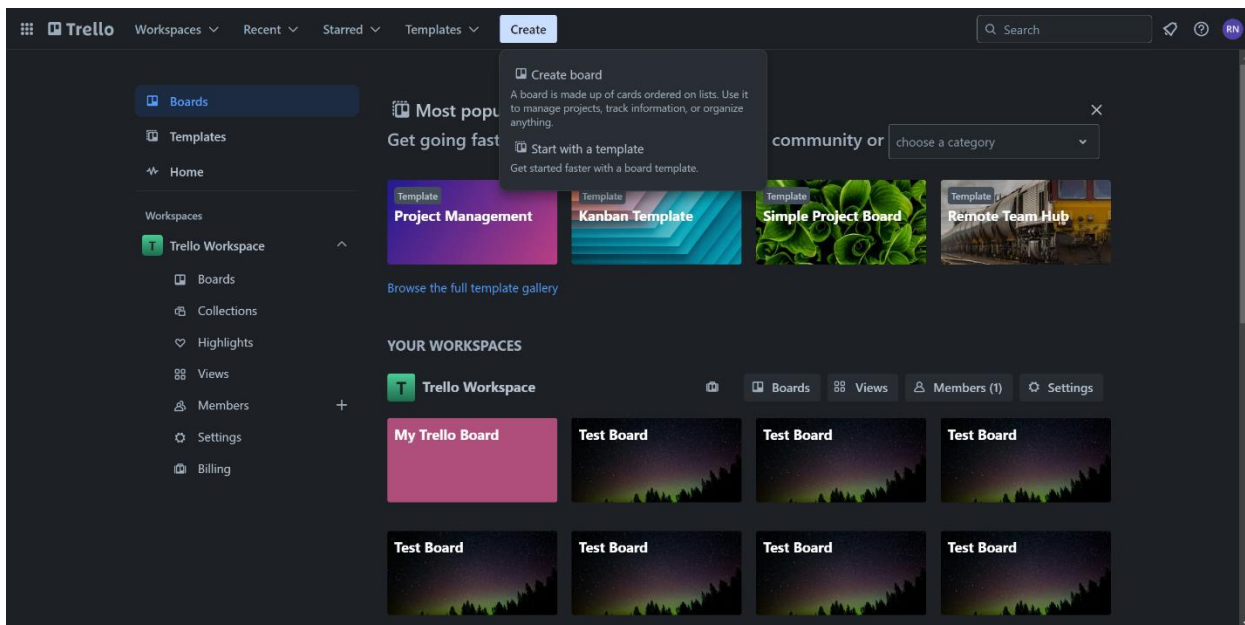

**Invalid Login Credentials**
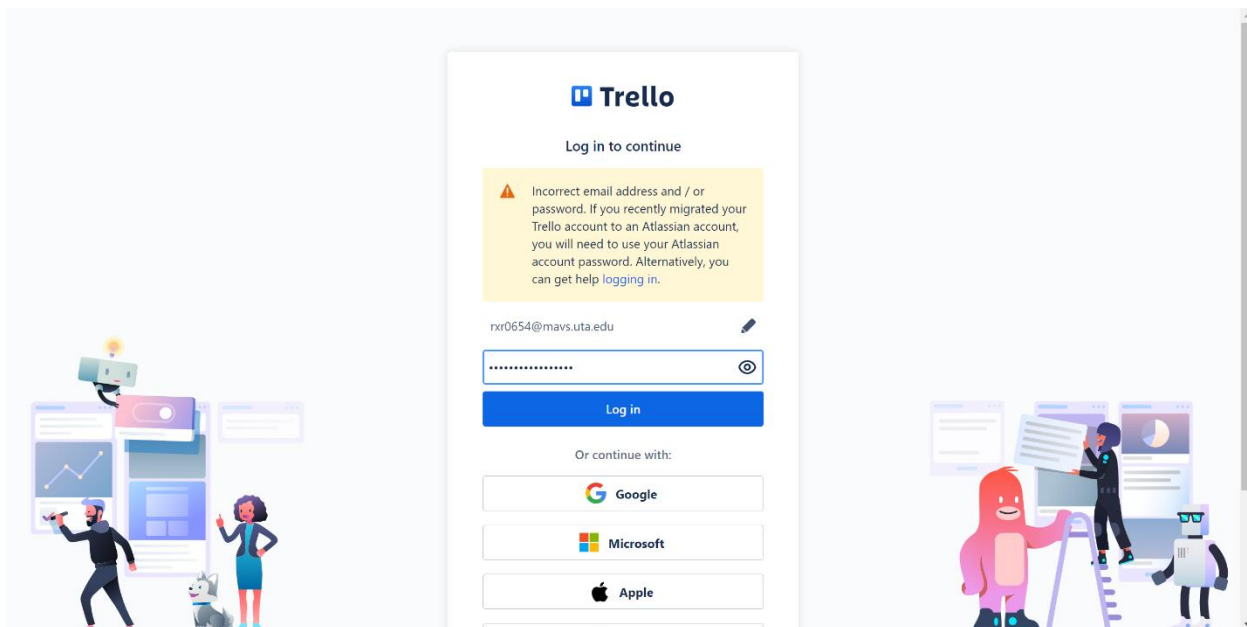
**2. Verify the creation of new boards, lists, and cards.**
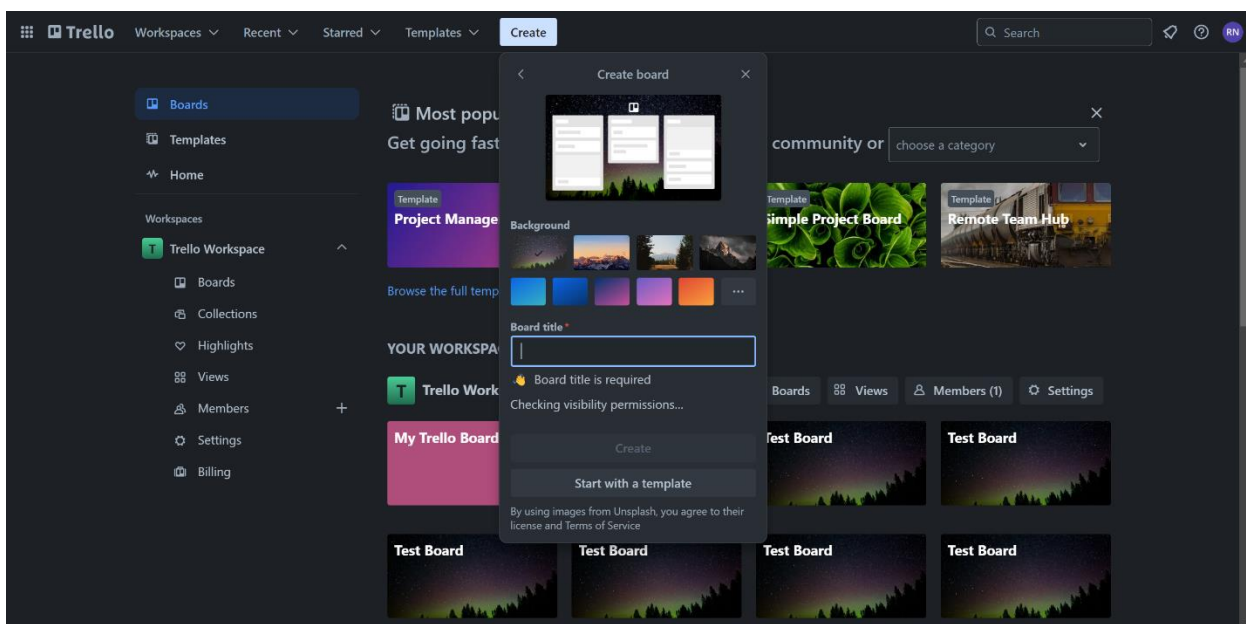   **Automate creation of new board, new list and new card after login**


Create Board
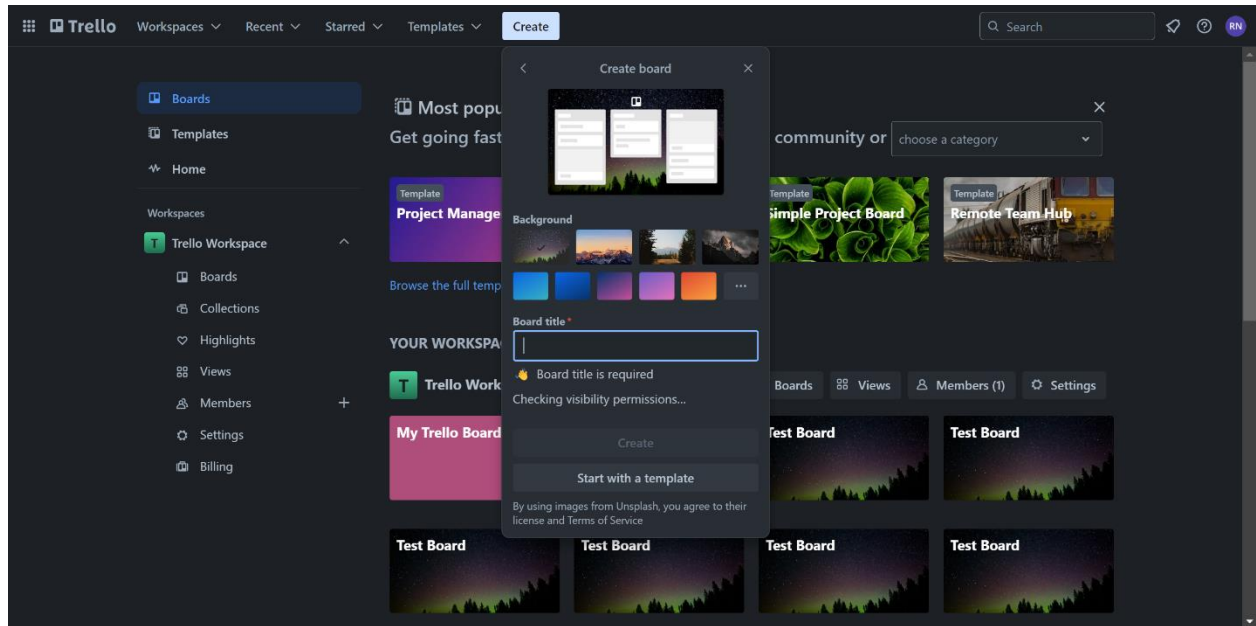
Test Data: Test Board for "Board Title"

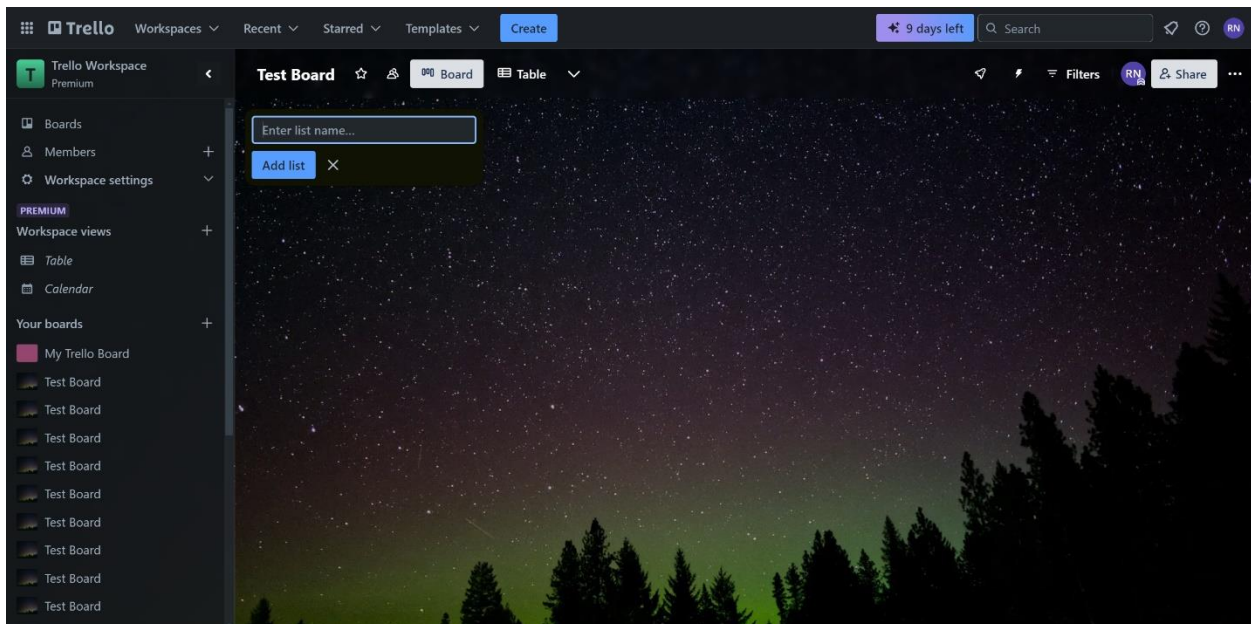 **Create** Button Clicked




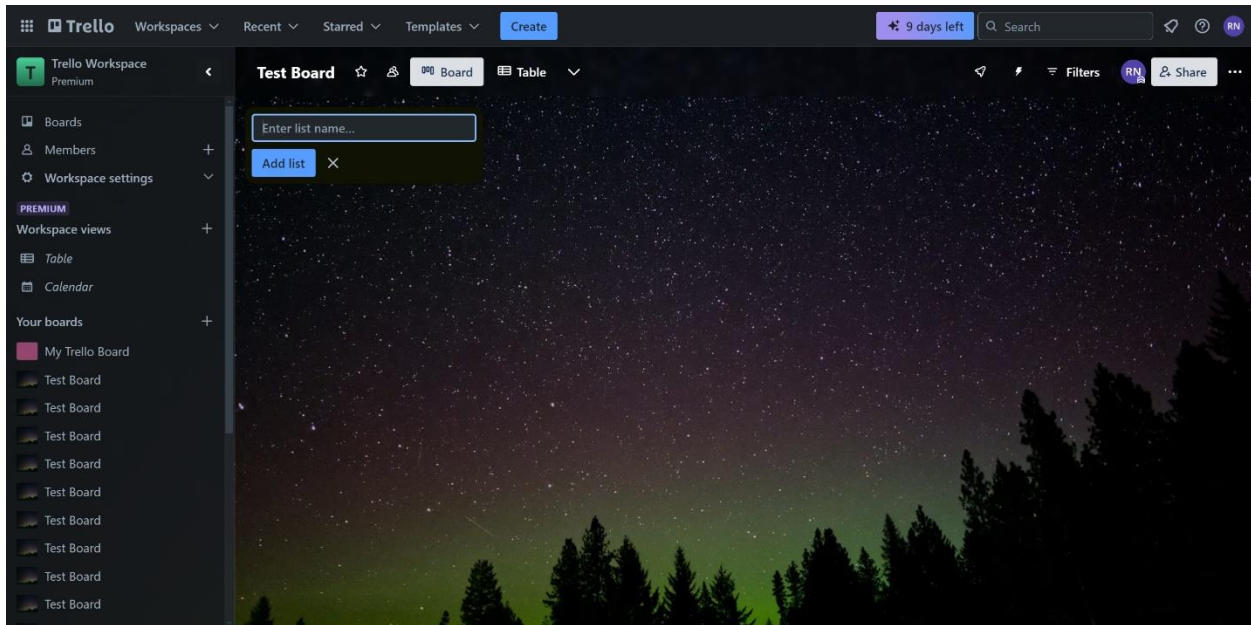   **Create Board** Button Clicked

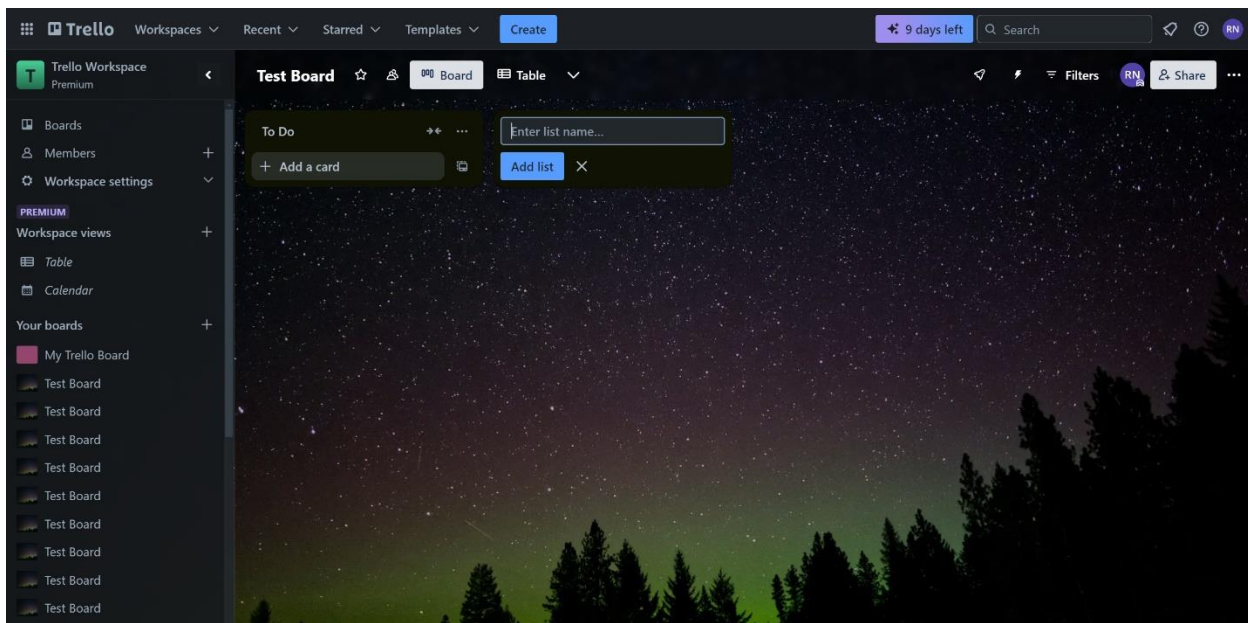Inputting Board Title



Board Created and Verified

## 2. Create List

**Test Data: Creating a list called 'To Do'**

Clicking on add a list



'To Do' List created

### 3. Verify drag-and-drop functionality for moving cards between lists. Automate the movement of moving a card from one list to another

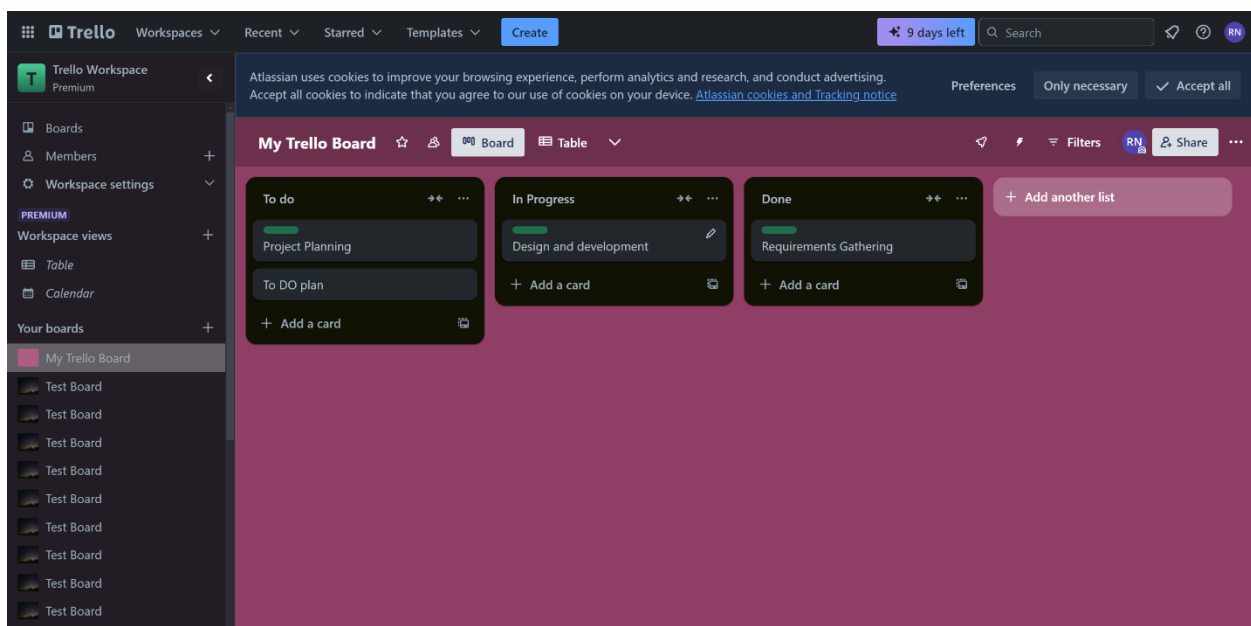Navigate to this main page 'https://trello.com/b/GH3SE4Yw/my-trello-board'

Assuming already loginned, else login functionality included.

**Test Case: Drag 'Design and development' from To Do List to In Progress List.**

Before Dragging and Dropping
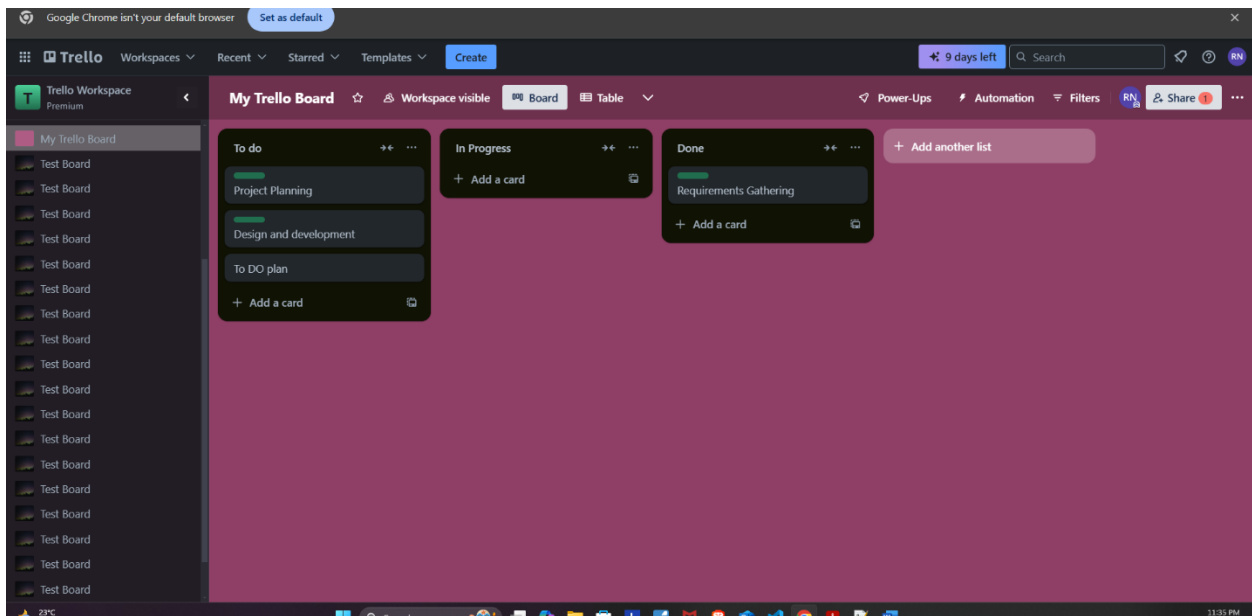


Drag and Drop Success

**4. Verify that specific elements (like buttons, links, and input fields) are correctly displayed on the dashboard**

**Automate navigation to main dashboard and identify and verify specific elements are displayed in the respective dashboard.**
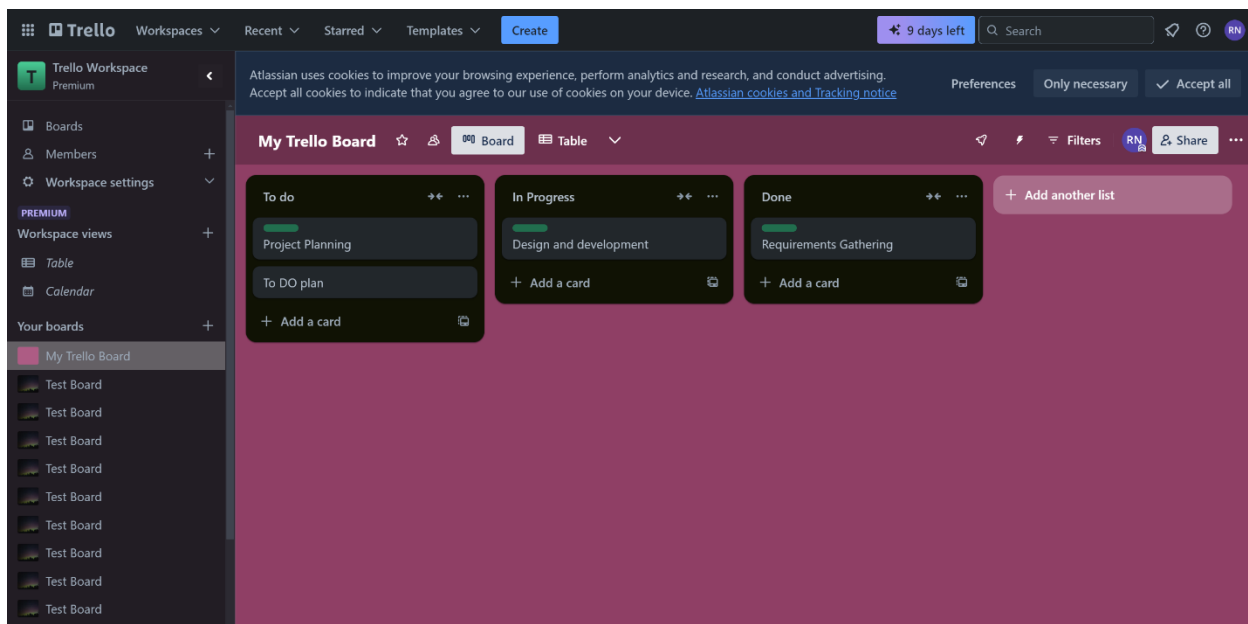
Navigate to this page: https://trello.com/b/GH3SE4Yw/my-trello-board

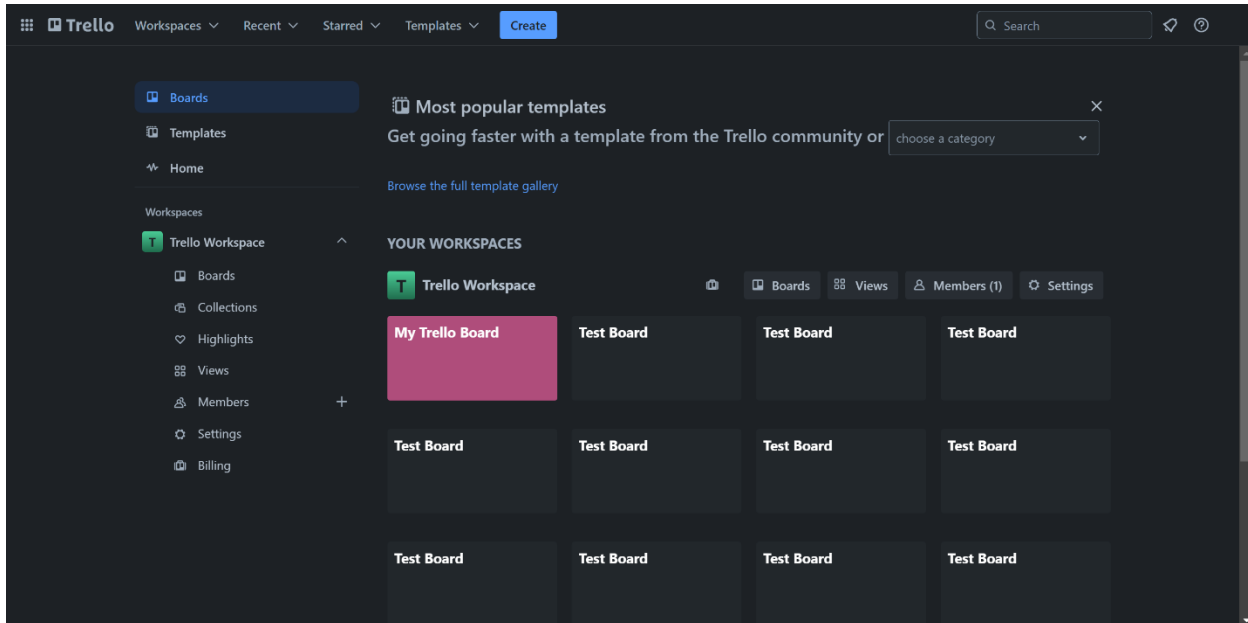It is assumed the user is logged in; else login functionality is also included.

**Verify 'add a card' button is visible**
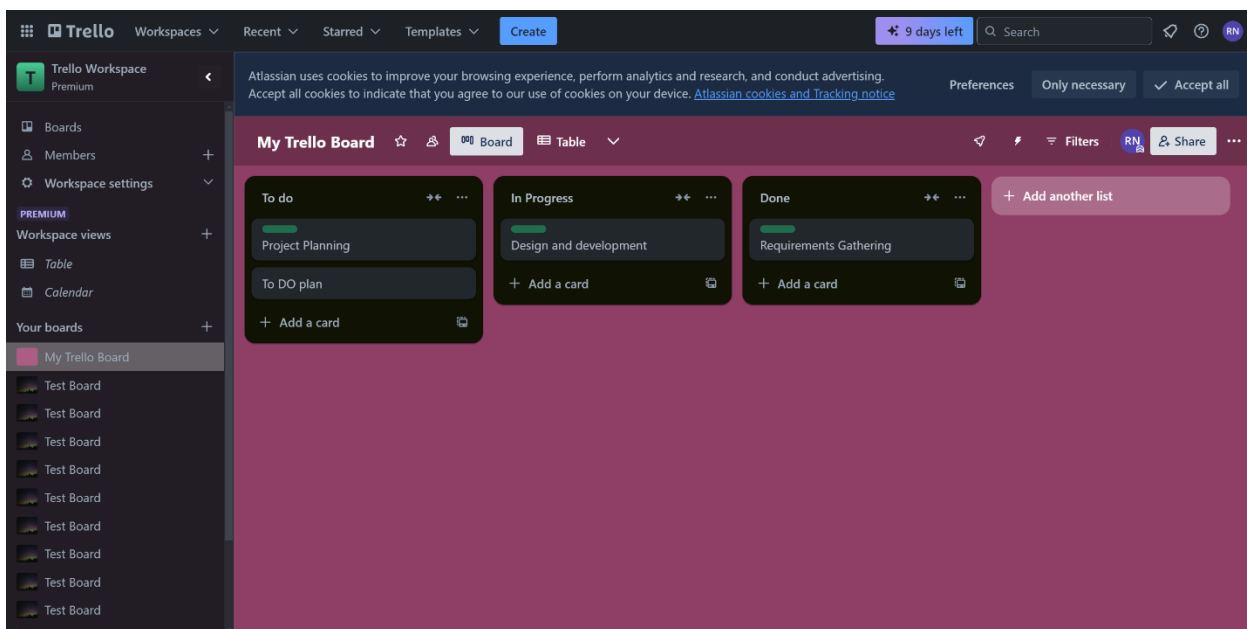


**Verify 'add another list' button is visible**

**Verify the search bar is present and can be interacted with**



## 5    Verify proper functionality of filtering, labeling, and search features

**Automate testing functionality of filtering using labels, labelling using custom labels to different cards and search using keyword in main search bar.**
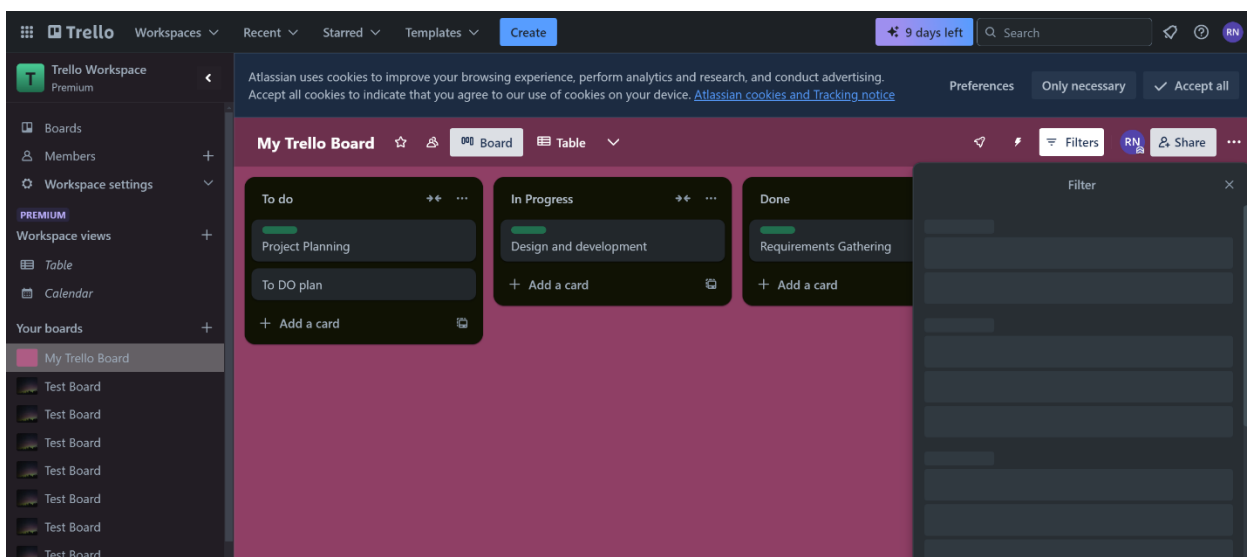
Navigate to the this page: https://trello.com/b/GH3SE4Yw/my-trello-board

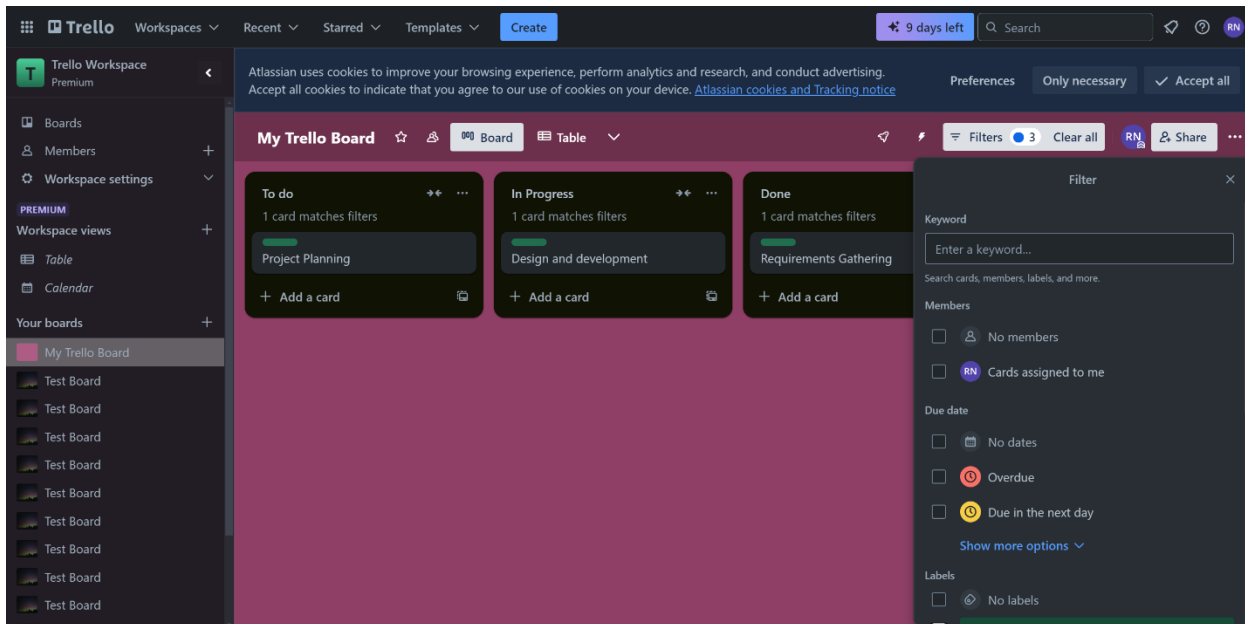It is assumed the user is logged in, else login functionality is also included.

**Filter**

**Test Data: Custom Green label after clicking on Filter button**
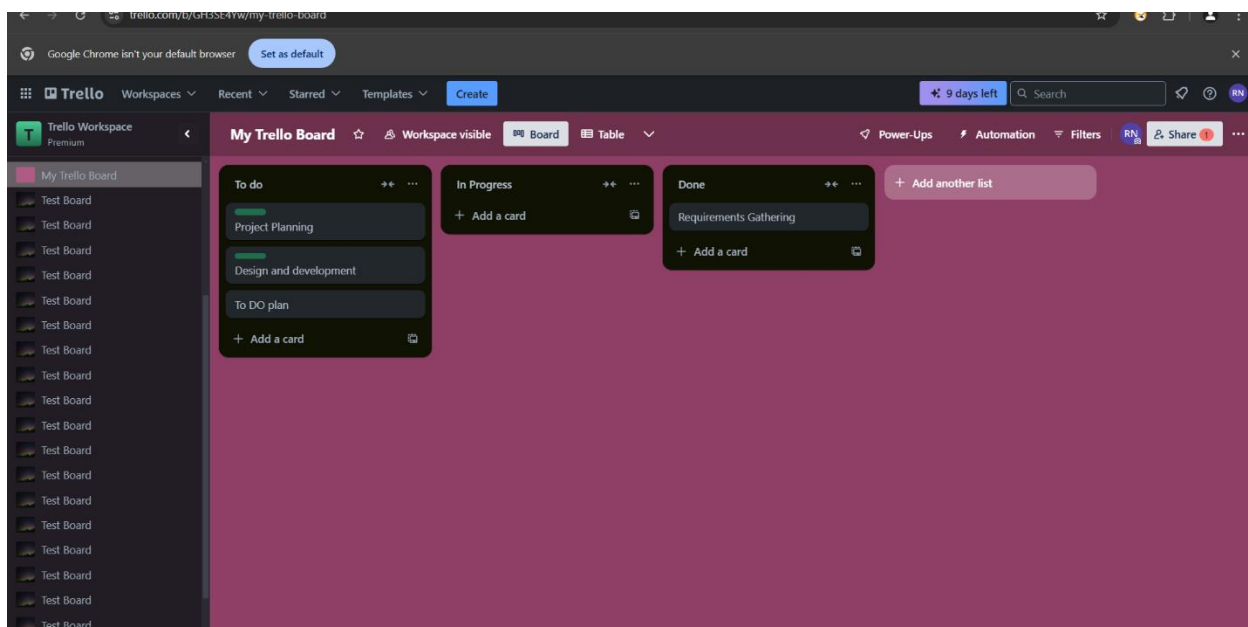
Clicking on filter button

Expanding the drop down and clicking on custom green label, and filtered card displayed.
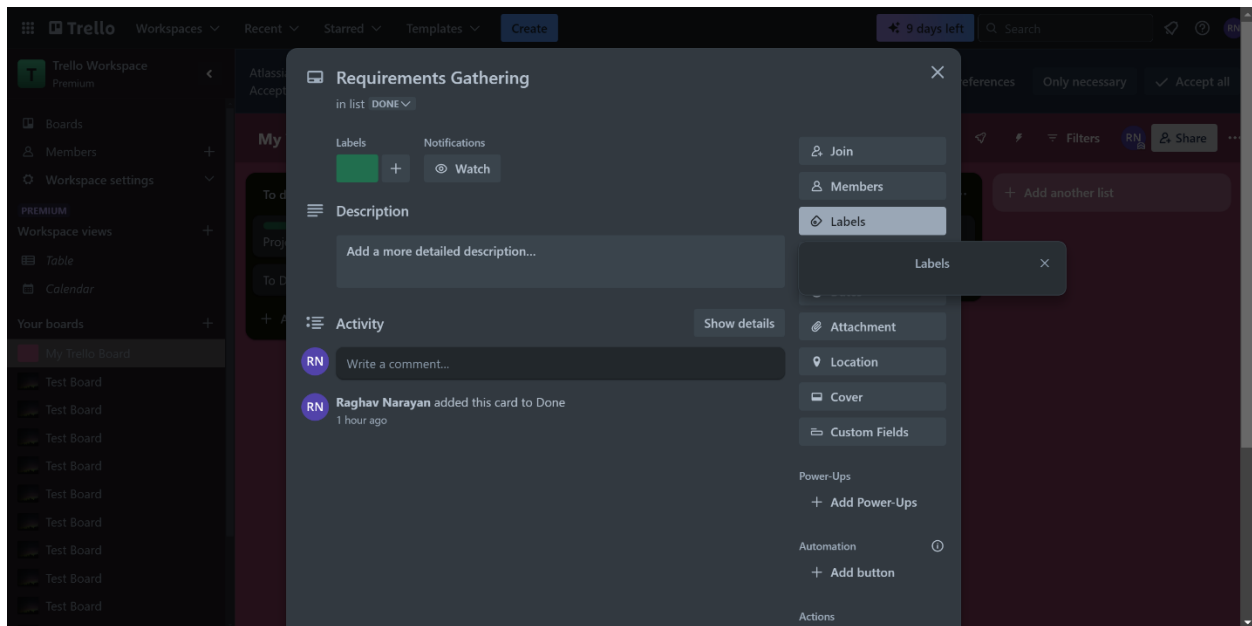


**Labelling**

Test Data and Scenario: Clicking on 'Requirements Gathering' card and adding a custom green label by navigating the modal component
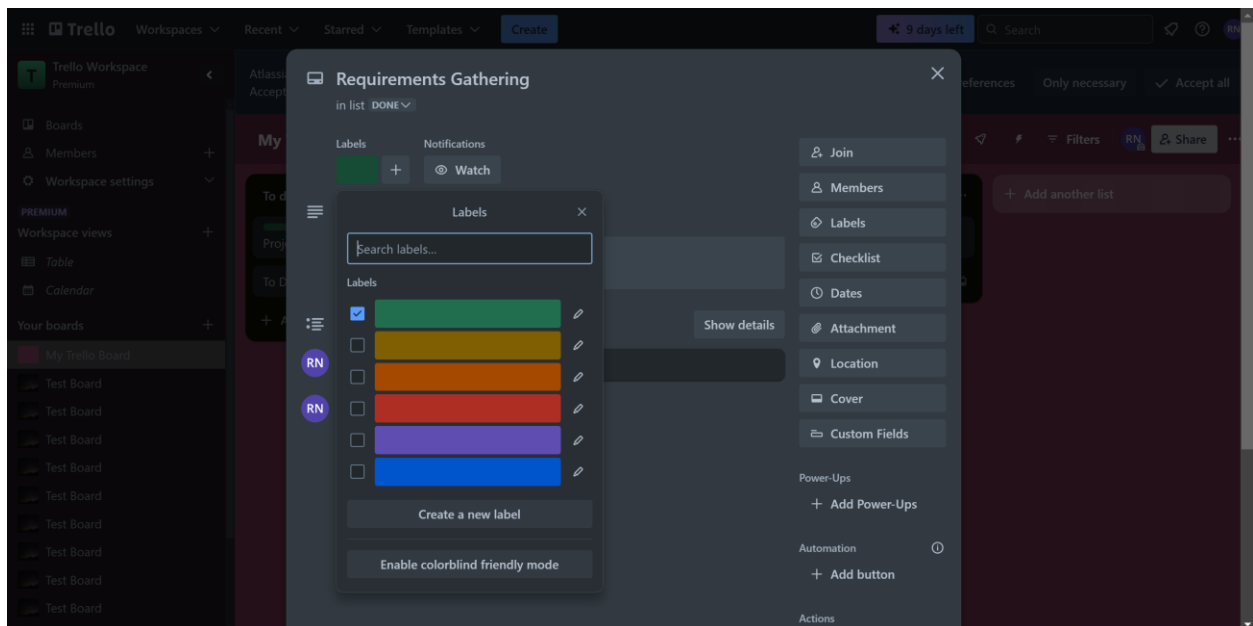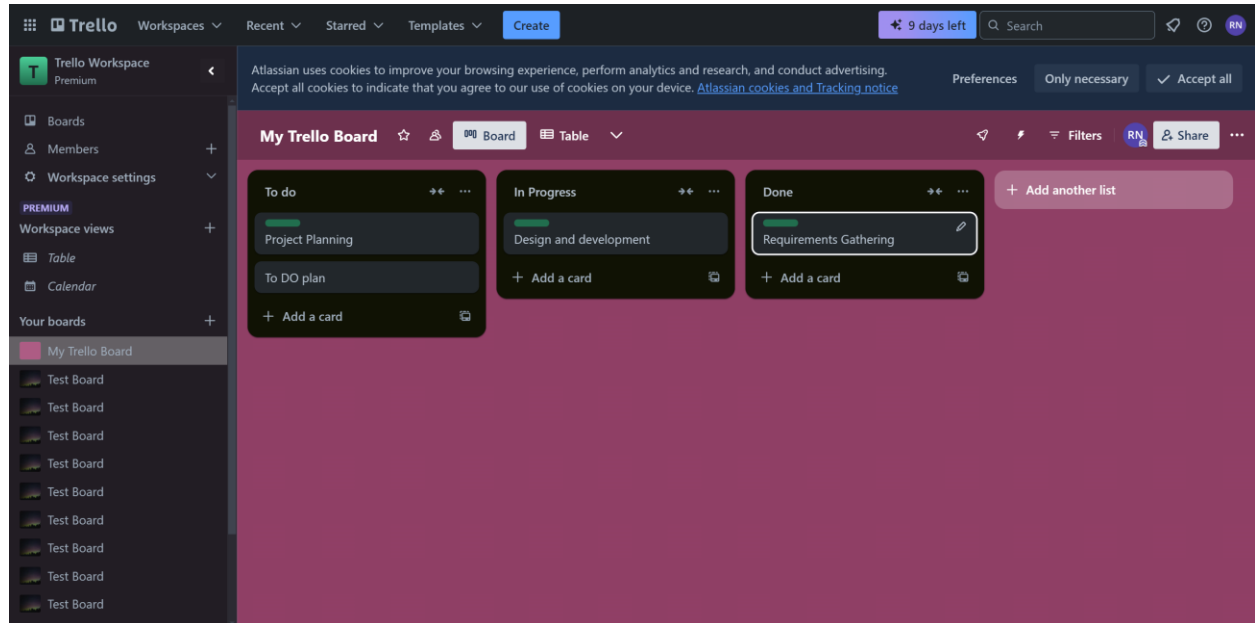
Home Page

## Labels Button Selected
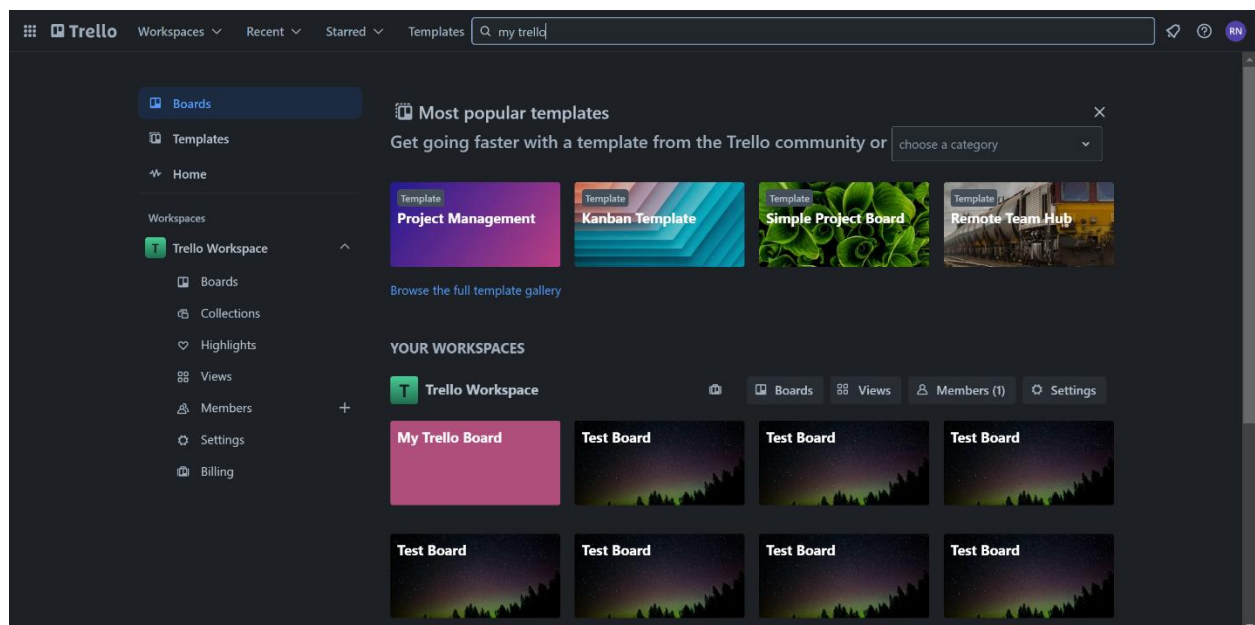


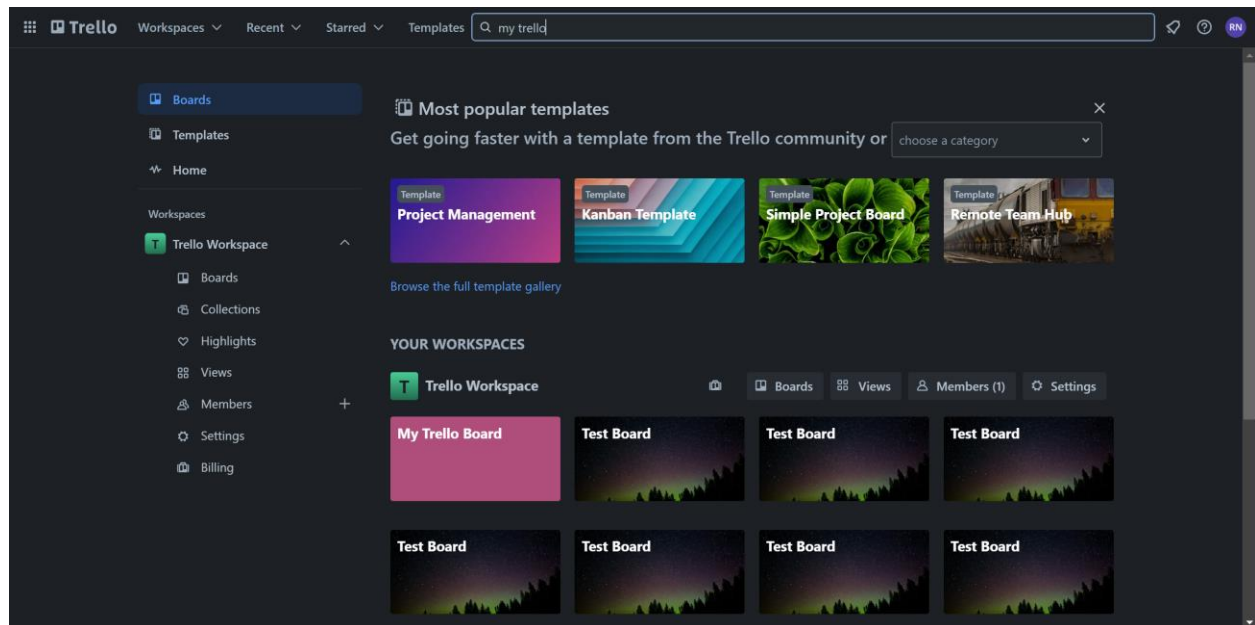## Selecting the Green Label

Green Label Verified



**Searching**

Test Scenario and Data: Navigating to home page, targeting the search bar and entering input as 'my Trello' to verify if list of boards appear in the workspace

Inputting the Search

Directed to intended workspace board



# Note:

Each script includes a screenshot functionality that allows the script to take screenshot of the viewport during each of the main steps in each scenarios. This will be included in the zip folder for reference.

Report. HTML also attached with the zip folder to get a summary of test results all the involved test cases.

Test Suite Coverage Also have been executed and respective test script-wise documentation report stored in respective coverage folder. Will be included in the zip folder.