Trello Board CRUD Operations Automated Test Plan

For

KZA Automated Testing challenge

Trishan Vassan

February 21 2022

Table of Contents

Introduction

This test plan will cover a series of manual test cases that covers the CRUD operations on Trello board Website and will cover both positive & negative cases. From these manual test cases an approach will be decided in this test plan to determine which test cases will need to be automated and whether they will be UI or Backend tests

Test Objectives and requirements

- Defining manual test cases that covers the CRUD operations on Trello board. Cover both positive & negative cases.
- Create test plan for the above created test cases. Approach to decide what needs to be tested at Backend & Front-End level.
- Perform CRUD operations on Trello board.
- Automate (at least three) of the above created test cases for both UI and Back-End.
- Provide a test report.
- Implement the solution using a BDD framework written in C#.
- Setup the solution from scratch. Code should follow OOPs and design principles.
- The repository must include clear instructions on where to find each deliverable and how to execute the tests.

Scope of Testing

The manual and automated tests of the Trello board application will include the Create, Read, Update and Delete Operations both manually through the Website UI and Postman and Automated through a BDD test Framework written in C#

System Overview

Trello is a collaboration tool that organizes a team's projects into boards and can let team members know what is being worked on, who is working on what and where a specific task is in a process. Trello consists of Boards, Lists and cards

Boards can be referred to the projects that teams are working on

Within each board there can be multiple lists (also known as swim lanes) and can have values such as "To Do", "In Progress", "Done" etc. or whatever unique values each team would like.

Cards are the tasks themselves e.g., "Set up testing framework" and the cards will move between the different lists depending on its status (To do, in progress, blocked etc.) and can be assigned to individual people, this is how teams can get a better overview on what work is being done, who is doing the work and in what state is the work/task in.

Approach

Assumptions/Constraints

Assumptions

- Trello's Frontend UI and API are working as expected and have no downtime during the testing

Constraints

- A few days may not be enough time to cover all possible testing scenarios, but the main CRUD operations will be covered

Test Coverage

Test coverage will be measured by:

- A completed test report indication all covered test cases for Frontend and Backend and the test results for those test cases

Requirements

In order to obtain the test objectives and deliverables, the following requirements are needed before any testing can begin:

- Manual Test Cases
- Trello Account to access Website UI
- Trello Account to obtain API Key and Token for Backend
- Computer/laptop capable of creating a BDD test framework in C#

Test tools

The following test tools will be used:

Manual Testing:

- Postman for backend testing
- Trello Website UI for frontend testing (chrome browser)

Automated Testing:

- Visual Studio
- Github
- Specflow + RestSharp for backend API testing
- Specflow + Selenium for Frontend UI testing

Test Reporting:

- Specfow livingdoc will be used for the test report generation

Features to be tested

Test Type based on Manual tests

From the Manual tests specified in the Manual Test Cases file of this project

This is how they will be organized between Backend & Front-End and whether or not it will be a manual or automated test:

Test Case	Backend or Frontend	Manual or Automated
General Test Cases		
Given a user has a Trello Account **When** the user attempts to login to the Trello Website using **valid credentials** **Then** the user should be logged in successfully **And** be able to view all boards the user has created and has been assigned to	Frontend	Manual and Automated
Given a user has a **valid Trello API Key and Token** **When** the user attempts to hit the Trello API **Then** a valid Response is returned without errors	Backend	Manual and Automated
Given a user that does not have a Trello Account **When** the user attempts to login to the Trello Website using **invalid credentials** **Then** the user will not be able to log in **And** will receive an error message stating that their credentials does not have an account	Frontend	Manual and Automated
Given a user does not have a valid Trello API Key and Token **When** the user attempts to hit the Trello API with **invalid API Key and Token** **Then** an a Unauthorized status and Error message is shown	Backend	Manual and Automated

Create/POST Test Cases		
Given a user with a Trello Account logs in successfully **When** the user attempts to create a **new board** **Then** the board should be created successfully **And** the user is able to access and view the new board	Frontend and Backend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user navigates to a board they have access to **And** the user attempts to create a **new List** within a list on the board **Then** the List should be created successfully in the board	Frontend	Manual
Given a user with a Trello Account logs in successfully **When** the user navigates to a board they have access to **And** the user attempts to create a **new Card** within a list on the board **Then** the Card should be created successfully in its specific list	Frontend	Manual
Given a user with a Trello Account logs in successfully **When** the user attempts to create a new board **without a name** **Then** the user receives a validation message **And** the board does not get created	Frontend	Manual and Automated
Given a user has a Trello API key and token **When** the user attempts to create a new board with an **invalid name** through the API **Then** a Bad Request status and Error response is shown **And** the board does not get created	Backend	Manual and Automated

Read/GET Test Cases		
Given a user with a Trello Account logs in successfully **When** the user attempts to view **all boards** they have access to **Then** the user is be able to access and view the assigned boards	Frontend and Backend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to view a **specific board** they have access to **Then** the user is be able to access and view the specific board	Backend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to view **all lists** on an existing board **Then** the user is able to view all lists on selected board	Backend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to view **all cards** on a valid list **Then** the user is able to view all Cards on selected board/list	Frontend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to search for a board they **do not have access** to **Then** the user is unable to view and access the specific board	Frontend	Manual
Given a user has a Trello API key and invalid token **When** the user attempts to retrieve a board through the API with an **invalid token** **Then** a Unauthorized status and Error response is shown **And** the board does not get returned in the response	Backend	Manual and Automated

Given a user has a Trello API key and token **When** the user attempts to retrieve a board through the API with an **invalid board id** **Then** a Bad Request status and Invalid Id response is shown **And** the board does not get returned in the response	Backend	Manual and Automated
Update/PUT Test Cases		
Given a user with a Trello Account logs in successfully **When** the user attempts to **update an existing board** they have access to **Then** the user is be able to access and update the specific board values **And** the board will be successfully updated with the new values	Frontend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to **update an existing list** on a valid board **Then** the user is be able to access and update the specific list values **And** the list will be successfully updated with the new values	Frontend	Manual
Given a user with a Trello Account logs in successfully **When** the user attempts to **update an existing cards** on a valid list **Then** the user is be able to access and update the specific card values **And** the card will be successfully updated with the new values	Frontend	Manual and Automated

Given a user with a valid Trello API Key and Token **When** the user attempts to **update an existing board** they have access to through the API **Then** the user is be able to perform a PUT request and update the specific board values **And** the board will be successfully updated with the new values	Backend	Manual and Automated
Given a user with a Trello Account logs in successfully **When** the user attempts to search for a board they **do not have access** to **Then** the user is unable to view and update the specific board	Frontend	Manual
Given a user has a Trello API key and invalid token **When** the user attempts to update a board through the API with an **invalid Board ID** **Then** a Unauthorized status and Error response is shown **And** the board does not get updated	Backend	Manual and Automated
Delete Test Cases		
Given a user with a Trello Account logs in successfully **When** the user attempts to create a **new board** **And** the board should be created successfully **And** the user is able to access and view the new board **Then** the user attempts to close and **delete the newly created board** **And** the user confirms that the board no longer exists	Backend and Frontend	Manual and Automated

Given a user with a Trello Account logs in successfully **When** the user attempts to create a **new list** on an existing board **And** the list should be created successfully **And** the user is able to access and view the new list **Then** the user attempts to close and **delete the newly created list** **And** the user confirms that the list no longer exists	Frontend	Manual
Given a user with a Trello Account logs in successfully **When** the user attempts to create a **new Card** on an existing board/list **And** the card should be created successfully **And** the user is able to access and view the new card **Then** the user attempts to close and **delete the newly created card** **And** the user confirms that the card no longer exists	Frontend	Manual
Given a user with a Trello Account logs in successfully **When** the user attempts to search for a board they **do not have access** to **Then** the user is unable to view and delete the specific board	Frontend	Manual
Given a user has a Trello API key and token **When** the user attempts to retrieve a board through the API with an **invalid board id** **Then** a Bad Request status and Invalid Id response is shown **And** the board does not get deleted	Backend	Automated

The above test cases were split between manual testing and automated testing as the automated testing cases covered all the CRUD operations required while the manual testing just ensure further that the CRUD operations were working as expected.

For the automated tests they are labeled as Manual and Automated as initially the manual test had to be done before implementing the automated test.

In terms of Backend coverage vs Frontend test coverage, I had ensured that had covered all 4 CRUD operations on both the Backend tests and the frontend tests

Pass/Fail Criteria

To pass the automated test, the following criteria must be met:

- All steps are executed and passed successfully
- No manual intervention has been performed during the test
- All test cases pass 100% without any errors

If any of the above criteria are not met the tests are automatically classified as a Fail

Exit Criteria

In order to achieve the exit criteria, a mandatory run rate of the automated tests is to be to be 100% unless a clear reason is given.

Test Results/reports

The test results and report will be included in the Test report section of the deliverables which will show each test cases and it's run time, pass, or fail status and scenarios covered

Installation and running of test

The installation and running of tests guide will be added into the Test repository's Readme.md file with a step-by-step guide to running the tests and generating test reports