TEST PLAN
FOR
AUTOMATION PRACTICE WEBSITE

Creator: Ly Cao
Created Date: July 12, 2024
Updated Date: July 15, 2024

Table of Contents

1. Introduction

- Automation Practice is a feature that provides pages containing many elements for automation testers to practice identifying and interacting with elements and writing automation scripts.
- The Test Plan has been created to communicate the test approach to team members. It includes the Test Objective, Scope, Risk-Based Testing, Testing strategy, Testing scenarios, Resources, Timelines and Conclusion.
- Since we are using the Scrum model to develop this project, this test plan will only cover current sprint.

2. Test Objectives

- The goal of this testing is to verify the functionality, usability, and overall user experience of the Automation Practice section. This includes validating exercise functionality, element identification clarity, and user interface effectiveness for automation testers.

3. Test Scope

- As planning meeting, this sprint focuses on the following areas of the Automation Practice section:

Page	User story	Sprint	Notes
Automation Practice page	Automation Practice page	Current sprint	
Complicated page	Section of buttons	Next sprint	Out of scope
	Section of Social Media Follows	Current sprint	
	Submit form 1 (1)	Current sprint	
	Submit form 2 (2)	Current sprint	
	Login form 1	Next sprint	Out of scope
	Login form 2	Next sprint	Out of scope
	Toggle form	Next sprint	Out of scope
	Navigation menu	Next sprint	Out of scope
	Comment	Next sprint	Out of scope
Fake Landing page		Next sprint	Out of scope
Fake Pricing page		Next sprint	Out of scope
Fill out forms page		Next sprint	Out of scope
Sample Application Life cycle page		Next sprint	Out of scope
Sign in page		Next sprint	Out of scope
Simple HTML elements page		Next sprint	Out of scope
Header page		Implemented from another sprint	Out of scope
Footer page		Implemented from another sprint	Out of scope
Security test		Tested by another team	Out of scope

- Test environment: This web application works on Chrome, Firefox, and Microsoft Edge.

4. Testing Approach

- ***Risk-Based Testing (RBT)***: Prioritize testing efforts on areas with high likelihood and impact of failure.
- ***Black-Box Testing***: Test the website from a user's perspective without knowledge of the internal code.
- ***API Testing***: Validate the functionality, performance, and reliability of APIs that the website interacts with.
- ***Ad-hoc Testing***: Conduct ad-hoc testing to identify potential issues not covered in specific test cases.
- ***Regression Testing***: Re-run a subset of critical test cases after significant code changes or new feature implementations to ensure existing functionalities remain intact (e.g. navigation links on the header and the footer).

5. Risk-based testing

- The High priority user stories will be implemented/ tested first

User story	Risk	Impact (5: High)	Likelihood	RBP Score	Priority
Automation Practice page		5	3	15	High
Section of Social Media Follows		3	2	6	Medium
Submit form (1) – on top of page	- Invalid User Input - The potential performance impact of running multiple test scripts concurrently highlights the importance of performance testing	4	3	12	High
Submit form (2)- on bottom of page	the same as Submit form 1	3	3	9	High

- Notes: RBP Score = Impact * likelihood

6. Testing scenarios

- In this test cases section, I have added an additional column named "Auto". Since these pages are designed to assist Automation Testers in practicing test script writing, we need to ensure that these elements can also be easily used for writing test scripts. However, depending on the availability of testers in the scrum team, we will only write automation scripts for those test cases that are marked with a "YES" value. For unimportant test cases, we will not write scripts and mark them with a "NO" value.

- Additionally, there will be test cases where automation test scripts can only be written to test a portion of the functionality, while the remaining part needs to be checked manually, such as verifying the position of elements on the GUI. In these cases, I will also set the "Auto" value to "NO".

- User story: Automation Practice page

Testing type	Summary	Priority	Auto?	Notes
Functionality	Verify that Automation Practice page loads successfully	High	No	**Expected result:** - Page loads successfully when clicking on Education > Automation Exercises menu. + Steps: 1. Access ultimateqa.com/automation/ 2. Click Education > Automation Exercises - Page loads successfully when entering

				https://ultimateqa.com/automation/ url on a web browser. - Page loads successfully when clicking on https://ultimateqa.com/automation/ link via email, apps. - Page load successfully when clicking on Back button on web browser after navigating to another page.
	Verify clicking on each navigation link successfully redirects the user to the corresponding page.	High	No	**Steps**: 1. Access this page 2. Click on 'Big page with many elements' **Expected result:** - System navigates to https://ultimateqa.com/complicated-page **Notes**: - Other links will be implemented on the next sprints
UX	Verify GUI displays as the design	High	No	**Steps**: 1. Access this page **Expected result**: - Title is "Automation Practice Use your skills to learn how to automate different scenarios" - 'Big page with many elements' link is visible - Front size, font color, align and position are the same as the design - Tab name displays: Automation Practice - Ultimate QA
	Verify website layout and content are optimized for different devices and screen sizes.	Medium	No	
Performance	Measure website loading speed across different devices and internet connections.	High	No	

	Conduct website load speed tests under simulated user traffic.	High	Yes	

- User story: Submit form (1), (2) on Complicated page

Testing type	Summary	Priority	Auto?	Notes
Functionality	Verify that Complicated page loads successfully	High	Yes	**Expected result:** - Page loads successfully when clicking on 'Big page with many elements' from Automation Practice page. - Page loads successfully when entering https://ultimateqa.com/complicated-page url on a web browser. - Page loads successfully when clicking on https://ultimateqa.com/complicated-page link via email, apps. - Page loads successfully when clicking on Back button on web browser after navigating to another page. - Page loads successfully after user logged in.
	Ensure that automation testers can uniquely identify and interact with Name field	High	Yes	**Expected result:** - Automation tester can create a script to verify that this field is present. - Automation tester can create a script to verify text of this field. - Automation tester can create a script to input data for this field. - Automation tester can create a script to change or clear data for this field.
	Ensure that automation testers can uniquely identify and interact with Email Address field	High	Yes	
	Ensure that automation testers can uniquely identify and interact with Message	High	Yes	

	field			
	Ensure that automation testers can uniquely identify and interact with Captcha fields	High	Yes	
	Ensure that automation testers can uniquely identify and interact with Submit button	High	Yes	**Expected result:** - Automation tester can create a script to verify that this field is present. - Automation tester can create a script to verify that this field is enabled or not. - Automation tester can create a script to verify text of this field. - Automation tester can create a script to click on this field.
	Verify that automation tester can submit successfully with valid data	High	Yes	**Steps:** 1. Access Complicated page 2. Enter valid data for all fields 3. Click Submit button **Expected result:** - Displays message "Thanks for contacting us" - Showing Login form
	Verify that error message show when automation tester leaves blank at Name, Email Address, Message, Captcha fields	High	Yes	**Steps**: 1. Access Complicated page 2. Don't input any data 3. Click Submit button **Expected result:** - Displays following message: "Please, fill in the following fields: . Name . Email Address . Message . Captcha"
	Verify that error message show when automation tester only input space (s) on	Low	Yes	**Steps:** 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter a space at all fields

	mandatory fields			4. Click Submit button **Expected result:** - Displays following message: "Please, fill in the following fields: . Name . Email Address . Message . Captcha"
	Verify that "Invalid email" message show when entering invalid data on Email address field	High	Yes	**Steps**: 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter invalid email address on Email field 4. Click Submit button **Expected result:** - Showing error Please, fix the following errors: ●Invalid email
	Verify that "You entered the wrong number in captcha." message show when entering wrong captcha number	High	Yes	**Steps:** 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter incorrect number in Captcha 4. Click Submit button **Expected result:** - Showing error "You entered the wrong number in captcha."
	Verify that red rectangle show on the field when entered invalid data on Name, Email Address, Message, Captcha fields	Medium	No	**Steps**: 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter invalid on Email field 4. Click Submit button **Expected result:** - Email Address field displays a red rectangle
		Medium	Yes	

	Verify that the error message will be updated after changing entered data and clicking Submit button.			**Steps**: 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter invalid on Email field 4. Click Submit button → Displays error message 5. Input valid data at Email field 6. Click Submit button **Expected result:** - The error message will be disappeared.
	Verify that entered data on page will be cleared after refreshing the page	High	Yes	**Steps:** 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter any data for all fields 4. Refresh page **Expected result:** - All entered data will be cleared
	Verify that Captcha values will reset after 30 minutes	High	No	**Steps:** 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' → take note data on Captcha fields 3. Waiting for 10 minutes > Refresh page → See actual result #1 3. Waiting for more 20 minutes > Refresh page → See actual result #1 **Expected result:** 1. Data on Captcha field still remains 2. Data on Captcha field is updated
	Verify that no error message show when entering long Name, Email, Message	High	Yes	
UX	Verify GUI of Complicated page displays as the design	High	No	**Steps**: 1. Access Complicated page

				Expected result: - Verify that page displays following texts: "Skills Improved: How to deal with a large page that has many elements and divisions? Properly structuring your page objects" - Front size, font color, align and position are the same as the design - Tab name displays: Complicated Page - Ultimate QA
	Verify GUI of Submit form under 'Section of Random Stuff' displays as the design	High	No	**Steps**: 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' **Expected result:** - Verify that 'Section of Random Stuff' displays as the design. - Verify that Name, Email Address, Message, Captcha and Submit button displays as the design. - Front size, font color, align and position are the same as the design - Tab name displays: Complicated Page - Ultimate QA
	Verify GUI of Login page displays as the design after submitting successfully	High	Yes	**Steps**: 1. Access Complicated page 2. Scroll down to 'Section of Random Stuff' 3. Enter valid data for all fields 4. Click Submit button **Expected result:** - Displays Submit form with the GUI is the same as the design
	Verify website layout and content are optimized for different devices and screen sizes.	Medium	No	
Performance	Complicated page	High	Yes	

	should load within a defined time-frame (e.g. less than 3 seconds).			
	Conduct Complicated page load speed tests under simulated user traffic.	High	Yes	
	Simulate multiple users submitting the form concurrently to evaluate server load and response times.	High	Yes	
	Measure the time it takes to submit the form with a large message field filled (e.g. 5000 characters).	Low	Yes	
	Measure the time it takes to submit the form when entering large message field filled (e.g. 5000 characters).	Low	Yes	

- User story: Section of Social Media Follows

Testing type	Summary	Priority	Auto?	Notes
Functionality	Verify clicking on each X icons successfully redirects the user to https://x.com/Nikolay_A00	High	Yes	**Steps:** 1. Access Complicated page 2. Scroll down to 'Section of Social Media Follows' 3. Click on the first X **Expected result:** - System navigates to https://x.com/Nikolay_A00 **Notes** - Repeat step #1 – step 3 for remaining X icons
	Verify clicking on each	High	Yes	

	F icons successfully redirects the user to https://www.facebook.com/Ultimateqa1/			**Steps**: 1. Access Complicated page 2. Scroll down to '*Section of Social Media Follows*' 3. Click on the first F **Expected result:** - System navigates to https://www.facebook.com/Ultimateqa1/ **Notes** - Repeat step #1 – step 3 for remaining F icons
	Ensure that automation testers can uniquely identify and interact with each X icon	High	Yes	**Expected result:** - Automation tester can create a script to verify that 5 X icons are present. - Automation tester can create a script to click on each icons correctly
	Ensure that automation testers can uniquely identify and interact with each F icon	High	Yes	**Expected result:** - Automation tester can create a script to verify that 5 F icons are present. - Automation tester can create a script to click on each icons correctly
UX	Verify GUI of 'Section of Social Media Follows' displays as the design	High	No	
	Verify website layout and content are optimized for different devices and screen sizes.	Medium	No	

7. Resources:

- To perform testing for this sprint, we need 2 QAs with the following qualifications and experience:

QA	Automation	Manual	API	Leader skill
Automation tester	Senior	Intermittent	Intermittent	Senior

Manual tester	Intermittent	Senior	Intermittent	Intermittent

- A testing framework (e.g., Cypress) to automate repetitive tasks.
- A tool for writing and monitor test cases
- A tool for tracking bugs (JIRA)
- Different web browsers and devices for compatibility testing (if applicable).

8. Testing Timelines:

Sprint	Timeline

Current sprint	• Day 1: Sprint planning
	• Day 1-5:
	  ◦ Write test cases & test data, write test scripts
	  ◦ Review test cases.
	  ◦ Setup test data
	  ◦ Setup test environment.
	• Day 6-8:
	  ◦ Execute Unit test, integration test
	  ◦ Find & report bug and verify fixed bugs
	  ◦ Update test cases/ test script if needed.
	• Day 7-11: System test, find bug, report bug
	• Day 8-12: Verify fixed bugs from Unit/Integration tests and perform regression test.
	• (Day 12: Review the backlog for the next sprint.)
	• Day 13-14: UAT testing, receive feedback.
	• Day 14:
	  ◦ Tracking bugs on production.
	  ◦ Retrospective then get lesson learn for the next sprint.
	  ◦ Review the backlog for the next sprint.

9. Testing Levels:

- Unit Testing: Focus on the functionality of individual website components (limited for this scenario).
- API Testing: Test individual APIs for functionality, performance, and error handling.
- Integration Testing: Verify how different website components and APIs interact with each other.
- System Testing: Test the website as a whole, ensuring all functionalities work seamlessly, including API integration.
- User Acceptance Testing (UAT): Involve real users to assess the website's usability and overall user experience.

10. Test Deliverables:

- In order to improve visibility and early risk identification, the Test Deliverables will be planned as below

Deliverable	Date/Milestone	Details	PIC	Report To
Test Plan Document	Sprint Planning	Comprehensive document outlining testing strategy, scope, approach, resources, and schedule.	QA Lead	Delivery Manager
Test Cases	Day 3 (End of Test Case Design Phase)	Detailed set of test cases covering all functionalities. Includes ID, description, steps, expected results, pass/fail criteria, priority.	Manual QA	QA Lead
Test Scripts	Throughout Development Cycle (as automation progresses)	Scripts in a testing automation framework (e.g., Selenium, Cypress) to automate test execution (optional).	Automation QA	QA Lead
Test Environment Setup Documents	Day 2 (Before Test Environment Setup)	Configuration and setup steps for the testing environment (hardware, software, network settings).	Infrastructure Team (or QA if responsible)	QA Lead
Traceability Matrix	Day 6 (After Test Case Design)	Matrix mapping requirements to corresponding test cases, demonstrating complete test coverage	QA Tester(s)	QA Lead
Test Execution Reports	After Each Testing Phase (Unit, Integration, System)	Reports summarizing test execution results, including passed/failed cases, identified bugs, and severity levels.	QA Tester(s)	QA Lead
Bug Reports	Throughout Testing Cycle (as bugs are identified)	Detailed reports for each bug, including ID, description, steps to reproduce, expected vs. actual behavior, severity, priority and related documents.	QA Tester(s)	Developer/ Development Team
User Acceptance Testing (UAT) Reports (if applicable)	After UAT Completion	Reports summarizing UAT results, user feedback, identified issues, and overall user satisfaction.	UAT Team (or designated testers)	Delivery Manager

Test Closure Report	End of Testing Cycle	Final report summarizing entire testing process, test coverage, identified/resolved bugs, overall results, and lessons learned.	QA Lead	Delivery Manager

11. Success Criteria:

- A minimum percentage of test cases pass (defined based on risk levels) across web functionalities, UX, performance, and API testing.
- Identified critical and high-severity bugs are resolved.
- The website functions as intended, providing a positive user experience.
- Regression testing ensures existing functionalities and API integration remain stable after changes.

12. Roles and Responsibilities:

- Test Lead: Manages the testing process, assigns tasks, and reports testing results.
- Manual Testers: Execute test cases for web functionalities, UX, performance, and user acceptance testing.
- API Testers: Design and execute API test cases for functionality, performance, and error handling.
- Automation Testers: designing and executing automated test scripts and report testing result to the test lead.
- Developers: Fix identified bugs and collaborate with testers on retesting.

13. Risks:

Risk Type	Risk Details	Solution
Requirement Risks		
Incomplete/Unclear Requirements	The requirements for the software might be incomplete, unclear, or poorly documented, leading to missed functionalities and potential issues.	- Conduct a thorough requirements review to ensure all functionalities are identified and clearly defined. - Work with stakeholders to clarify ambiguities and ensure requirements are documented comprehensively.
Missing Requirements	Critical functionalities might be entirely missing from the documented requirements, potentially causing major issues in the final product.	- Implement a strong communication channel between testers, developers, and stakeholders. Regularly review requirements to ensure they reflect the latest needs and identify any missing

		functionalities early on.
Conflicting Requirements	There might be conflicting or contradictory requirements documented, leading to wasted effort and potentially failing tests.	- Facilitate discussions between stakeholders to resolve any conflicts or inconsistencies in the requirements. Ensure all parties involved have a clear understanding of the desired functionalities.
Changing Requirements	Requirements might change during the development process, causing rework for testers and potentially leading to missed bugs.	- Implement a change management process to track and communicate requirement changes effectively. Testers should be involved in discussions about requirement changes to ensure test cases are updated promptly and testing efforts remain aligned with the evolving needs of the software.
Unrealistic Requirements	Requirements might be unrealistic or technically infeasible to implement, leading to wasted effort and project delays.	- During the requirement definition stage, involve testers and technical experts to assess the feasibility of proposed functionalities. This can help identify and address unrealistic requirements early on.
Functionality	Critical functionalities might not be thoroughly tested due to incomplete test case coverage.	- Conduct a thorough requirements review to ensure all functionalities are identified. - Develop comprehensive test cases covering all critical and high-priority functionalities. - Utilize exploratory testing techniques to identify any missing functionality.
Regression	New features or bug fixes might introduce unintended regressions in existing functionalities.	- Implement a strong regression testing suite that covers core functionalities. - Automate critical regression test cases for efficient execution. - Conduct smoke testing after each development build to identify major regressions.
UX	- The user interface might be complex or unintuitive, leading to	- Conduct usability testing with target

	user frustration. - The web application might not function correctly on all supported browsers, devices, or operating systems.	users early in the development cycle. - Clearly define supported platforms and devices in the test plan. - Develop responsive design elements to ensure a good user experience across various screen sizes.
Performance	The web application might experience slow loading times or instability under high user loads.	- Conduct performance testing to measure response times and resource usage. - Optimize code and database queries to improve performance. - Implement load testing to simulate real-world usage patterns and identify scalability bottlenecks.
Environment	Inconsistencies between development, testing, and production environments might lead to unexpected behavior in production.	- Clearly define and document the configuration of all testing environments. - Automate environment setup procedures to ensure consistency. - Conduct environment verification before deploying new builds to production.
Resource Constraints	Insufficient time, budget, or personnel resources might hinder thorough testing.	- Prioritize testing activities based on risk and impact. - Utilize automation tools and techniques to improve test efficiency. - QA members can hand overs tasks each others. - Manage stakeholder expectations by communicating potential limitations due to resource constraints.
Communication Gaps	Lack of communication between developers, testers, and other stakeholders could lead to misunderstandings and delays.	- Establish clear communication channels and protocols. - Conduct regular meetings to discuss testing progress and address any issues. - Document testing activities and results thoroughly.