Table of Contents

Automation Testing Strategy

1. Objectives:

- Automate test cases that are stable, highly repeatable, and easy to maintain.
- Optimize time and effort by focusing on critical test cases with high execution frequency.
- Ensure system quality by detecting defects quickly and accurately.

---

2. Tool Selection:

- **Katalon Studio**:
  - A user-friendly tool for both manual and automation testers, supporting visual test case creation.
  - Integrated features like Object Repository management, detailed reporting, and built-in keywords.
  - Supports multiple platforms: Web, API, and Mobile.

---

3. Criteria for Selecting Test Cases for Automation:

3.1 High Priority:

- Test cases that verify critical features, such as the Search functionality.
- Scenarios that directly impact user experience, like displaying accurate search results or error messages.

3.2 Reusability:

- Test cases executed frequently across different releases.

- For example, verifying the UI or validating search results with valid/invalid keywords.

3.4 Ease of Automation:
- Test cases with clear steps and easily defined expected outcomes.
- For example, entering keywords and verifying the displayed results.

3.5 Stability:
- Areas of the system that are less prone to changes or errors caused by UI or logic modifications.
- For example, the search result window and navigation functionality.

4.Selected Test Cases for Automation:

Based on the criteria, the following test cases are selected for automation:

TC_1: Search with valid keyword.

- **Justification:** This is a primary scenario to ensure that the search functionality works correctly.

TC_2: Search with invalid keyword.

- **Justification:** To assess the system's error handling capabilities when users enter invalid keywords.

TC_3: Verify navigation when selecting a result from the search result window.

- **Justification:** To ensure the navigation to the correct detailed page works properly when a user selects a result from the search result list.

---

5. Automation Implementation Strategy:

Environment Setup:

- Use **Katalon Studio** for test case creation and UI element management.
- Configure **Global Variables** to manage URLs and configuration details.

Object Repository Management:

- Create Test Objects for each UI element, such as:
    - Search bar.

    - Search result popup.
    - Result items in the list.

Test Case Development:

- Utilize Katalon Studio's built-in keywords, such as:

    - WebUI.openBrowser() – Open the browser.
    - WebUI.setText() – Enter keywords into the search bar.
    - WebUI.click() – Select a result from the list.
    - WebUI.verifyElementText() – Verify the search results.

Script Maintenance:

- Implement **Data-Driven Testing** to validate multiple datasets (valid/invalid keywords).
- Regularly update Test Objects and scripts in case of UI changes.

Reporting and Analysis:

- Use Katalon Studio's automated reporting to track test results.
- Periodically review and evaluate the effectiveness of the automated test cases.

Expected Outcomes:

- Accelerated testing process and reduced manual errors.
- A reusable and maintainable suite of automated test cases.
- Assurance of stable search functionality across multiple releases.