WireApps

Intern Quality Assurance Engineer

Technical Assessment

Test Automation Basics

W.G.P.Shehani

3. Test Automation Basics

Assume you are asked to create a test automation framework for a web application of an online shopping site. The application includes functionalities such as user login, registration, product search, add to cart, checkout, and order history etc...

What is your approach? (No need to write scripts/codes)

Test Automation Framework

1. Understand the Requirements and Define Scope

Identify and prioritize the key functionalities to be automated:

- User login and registration
- Product search
- Adding products to the cart
- Checkout process
- Order history

Determine the test cases that will be automated, focusing on repetitive tasks and those critical to the site's core functionalities.

2. Select Appropriate Tools and Technology

Choose tools that best fit the application and technology stack:

- Selenium WebDriver for web automation.
- TestNG or JUnit for test management and reporting.
- Maven or Gradle for build management and dependency management.
- Jenkins or GitLab CI for continuous integration (CI).
- Use a logging tool like Log4j for capturing logs and debugging.

3. Design the Test Automation Framework Architecture

- Use the Page Object Model (POM) design pattern to keep the test scripts modular, maintainable, and reusable. Each web page will have its own class representing the elements and interactions on that page.
- Create utility methods for commonly used actions such as login, navigation, and handling UI components like alerts and forms.

4. Organize the Folder Structure

Establish a clear folder structure for the test framework:

- Test Scripts: Store all the automated test cases.
- Page Objects: Create classes for each page of the application.
- Test Data: Store data in external files (Excel, CSV, JSON).
- Utilities: Shared functions and utilities.
- Reports: Logs and reports generated by the tests.

5. Implement Custom Logger

Use Log4j or another logging library to track test steps, log failures, and assist in debugging issues. A good logging mechanism is critical for diagnosing issues during test failures.

6. Create and Organize Test Cases

Organize test cases by the following categories:

- Smoke Tests: Basic functionality tests like login and registration to ensure stability.
- Regression Tests: Re-run tests for all major features after each release or update.
- End-to-End Tests: Test complete workflows, such as login, search, add to cart, checkout, and order history.

Implement data-driven testing to handle multiple data sets, ensuring broader test coverage for different input combinations.

7. Set Up Test Data Management

- Store test data in external files such as Excel, CSV, JSON, or a database, making it easy to update test cases without changing the scripts.
- Implement data-driven testing by feeding multiple inputs from external sources to validate different scenarios (eg. testing multiple login credentials or product search inputs).

8. Integrate with CI/CD Tools

- Integrate the automation framework with a CI/CD pipeline using tools like Jenkins or GitLab CI to run automated tests upon each code commit.
- Set up automatic notifications and reports to be sent to the development and testing teams when tests fail or succeed.

9. Create Test Execution and Reporting Mechanism

Implement reporting tools such as Allure or Extent Reports to generate detailed and visual test execution reports.

Ensure reports include:

- Test status (pass/fail)
- Logs
- Screenshots on failure for debugging purposes.

10. Review and Maintain the Framework

- Continuously review and update the framework to accommodate changes in the application or test cases.
- Ensure the framework is scalable and easy to maintain for future updates by adhering to coding best practices and ensuring the modularity of the codebase.

This approach ensures that the automation framework is robust, maintainable, and capable of providing comprehensive test coverage for all critical functionalities of the online shopping site. It prioritizes reusability, scalability, and clear reporting, which are essential for a long-term automation solution.