Remote software tester

Automation plan

Candidate: Stephen Dawn

Date: 12/07/2021

Overview

Normally, the considerations as to whether test automation is required at the functional level are dependant on the coverage in unit tests of the codebase but as this is being done purely as a black box testing exercise with no requirements then the considerations need to be somewhat different. There are some assumptions being made with this plan around the scope of testing and the completeness of the system to be tested. The tests listed here are designed more to demonstrate the thinking around this decision process rather than be an exhaustive list.

Assumptions

For the benefit of this piece of work, it is assumed that the only part of the system which is function complete and therefore suitable for automated testing is the energy buying screen. It is assumed that the web objects have been identified in their final state and references to them will be honoured if changes are made.

From the HTML observable in the developer console, it is also apparent that the same function is called from all forms and so only one will be tested. It is assumed that this behaviour will not change and the same function will continue to be called for each form.

Candidates with reasoning

1. A valid purchase should pass
    a. This is the basic happy path and should constitute the minimum automated testing.
2. Valid purchase reduces available stock
    a. Should be a simple validation and this failure could cause issues with more being sold than is available impacting customer experience and company reputation.
3. Zero entry in amount should fail.
    a. This is because the customer should never knowingly buy zero units and therefore any attempt would be accidental. A recommendation would be to remove the ability to click unless a value was entered or for the request to be ignored.
4. More than available should result in an error.
    a. This is critical for ensuring that more stock is not sold than is available. Further requirements should be sought for the correct handling of this but it is assumed here that an error is sufficient.
5. A negative entry should result in an error.

   a. A negative entry is unlikely to be accidental and could be malicious and designed to corrupt the database. This could affect other customers and could result in serious issues for the company.
6. NaN entry should result in an error.
   a. While this is currently the case, the key here would be to validate the copy on the error page and to see a **GO BACK** button for returning to the buy screen. This would also cover off any other errors such as the out of bounds error for numbers larger than an integer.

In addition to these, it is prudent to include a passing test for each energy type to ensure there have been no links broken. This would add a level of confidence that the system has not been broken inadvertently. It is not possible to do this for all test systems though as the Nuclear power option cannot be purchased at present.
It may also be beneficial to test that the values being displayed in the sale confirmation page are correct but this is out of scope for the automation at this stage.

Identifying elements

There are a number of methods that could be used to identify elements but many of these are recorders that are based on the javascript identifying the element itself and do not always take into account the parent elements. This means that the recorder may not access the specific element required as the only identifier for the **BUY** button is the same for them all irrespective of fuel type.

For this reason, it is quicker and easier to view the source code directly using the developer console which highlights the element and where it lives in the hierarchy. This allows us to determine the specific instance of the button to be actioned.

In the case of tests for Gas, this would require selecting the row where the element with *id="energyType_energyTypeId"* AND *value="1"*