Mandatory Assignment - Cypress Test Automation

1.3 Write a manual step-by-step test case for the “use customer care page” - scenario

Test Case ID: Customer Care
Test Case Name: Parabank Customer Care Form

Test Objective: I want a user to be able to submit a customer care form, stating issues or questions the user may have, and receive help as soon as possible

Preconditions:

- Connection to the internett
- Have a valid email address
- Have a valid phone number

Test Steps:
Step 1: Go to the website https://parabank.parasoft.com/parabank/index.htm
Expected Result: The user should be redirected to the home page of ParaBank

Actual Results: The user was redirected to the home page of ParaBank

Step 2: Click the orange mail box icon under the header *Welcome to ParaBank*
Expected Result: The user should be redirected to ParaBank customer care form with empty fields ready to be filled in

Actual Results: The user was redirected to ParaBank customer care form with empty fields ready to be filled in

Step 3: In the name field, enter your name
Expected Result: The field should be available for editing and the name should show

without any problem

Actual Results: The field was available for editing and the name was shown without any problem

Step 4: In the email field, enter your email
Expected Result: The field should be available for editing and the email should show without any problem

Actual Results: The field was available for editing and the name was shown without any problem

Step 5: In the phone field, enter your phone number
Expected Result: The field should be available for editing and the phone number should show without any problem

Actual Results: The field was available for editing and the name was shown without any problem

Step 6: In the message field, write your message
Expected Result: The field should be available for editing and the message should show without any problem

Actual Results: The field was available for editing and the name was shown without any problem

Step 7: When ready, press the button “Send to customer care”
Expected Result: The user should be redirected to a page saying that a customer care representative will contact them

Actual Results: The user was redirected to a page saying that a customer care representative will contact them

Passing Criteria:

- The servers of ParaBank are operating and working as expected

- The user is connected to wifi
- The user enters valid input into the form fields
- The user presses the button “SEND TO CUSTOMER CARE”
- There were noe interruptions, or errors from the client or cloud side while executing this workflow

Notes

The customer care form could be improved significantly. The form only requires the the length of the input for each field is greater than 0 and it doesn’t sanitise the value of what is give to the form. This means that a user can enter white space for each field and then be able to submit the form. In addition, since the form doesn’t even check what’s given, it is probably also not critical about what comes it, making the form vulnerable to attacks such as SQL injections or Server overloading.

Task 2

2.1 What other benefits do we get by ensuring that our automated tests are independent of each other?

In programming, it is generally more beneficial to write code with as few dependencies as possible. This practice is called encapsulation, and it enables each component of a system or test to have full control over its domain without relying on other components or tests to run as intended. Components or tests that are heavily dependent on others are more prone to failure, as they not only depend on themselves to pass, but also on other components and tests. If something changes that causes one test to fail, it could cause a ripple effect that could potentially affect the entire project. Fixing each of the components and tests would also take longer. Therefore, the more external dependencies there are, the higher the risk of having multiple tests fail.

Writing independent tests has the added benefit of avoiding redundant testing. When two tests that cover different parts of the system both rely on another test, running that test twice wastes resources. This is just the smallest example of unnecessary resource

usage. If we scale it up and imagine that 1000 tests depend on the same test, it becomes clear that this approach is wasteful.

2.2 Manually write three test cases for ParaBank

Test Case 1: Login to an account

- Test Scenario: Verify that the account exist and that the user can log in properly
- Test Data: CypressUser
- Expected Result: The user is able to login

Test Case 2: Toggle JMS service

- Test Scenario: Verify that the JMS service toggles successfully
- Test Data: No testdata
- Expected Result:
  - JSM service change state when prompted.

Test Case 3: Forgot login info

- Test Scenario: Verify that the user is able to recover their account
- Test Data: CypressUser
- Expected Result: The user is able to recover their account