Develop a test approach and document most critical bugs

Resource: [https://buggy.justtestit.org](https://buggy.justtestit.org/)

Outcome:

a short description of the test approach for this project

documented bug reports on 3-5 most critical bugs in the application

Automate the top 5 critical functionalities of BuggyCars. You may use any language, tool/framework and your solution should reflect your level of expertise.

**Test Approach:**

**Analysis:** Review the project and analyze the proposal of the project to clarify the goal and scope of the test. This step would ensure that the testing team has a clear understanding of the objectives of the project and what is expected from the testing process.

**Requirements Review:** Based on the analysis, review the project requirements, and identify any ambiguities or inconsistencies. This would ensure that the team has a clear understanding of what the application is supposed to do.

**Test Planning:** Based on the requirements, a comprehensive test plan will be created, outlining the scope of testing, test objectives, test strategies, and test methodologies to be used for the project. This would also involve the identification of test cases, test data, and test environments.

**Test Execution:** This stage involves the actual testing of the application using the defined test cases and test data. Testing would be done through automation.

**Defect Reporting and Management:** Any bugs or issues identified during testing will be reported in a defect tracking system, and the development team will be informed of the issues so that they can fix them.

**Test Closure:** Once all testing activities are completed, a test closure report will be prepared, summarizing the testing activities performed, test results, and any issues found. This report will be shared with the development team and other stakeholders.

1. **Analysis**

**Application Analysis**

BuggyCars is a web application that allows users to view and rate various buggy cars, providing a platform for users to analyze and vote on different aspects of the car. The application aims to simplify the process of finding the perfect buggy car by providing a centralized location for users to find information and connect with other enthusiasts.

**Test requirements analysis**

The purpose of this test requirements analysis is to define the test scenarios and test cases that will be used to ensure the BuggyCars application meets its intended purpose. The test requirements analysis will be used as a reference for developing and executing test cases to validate the functionality and behavior of the BuggyCars application.

**Scope:**

The scope of this test requirements analysis includes the following features of the BuggyCars application:

* User Registration
* User Login
* Buggy Car Navigation Functionality
* Commenting System (Voting System)
* Buggy Car Ranking System

The risk matrix analysis is a process of evaluating the impact and likelihood of risks associated with the critical functionalities of the system. The analysis helps in identifying the top 5 critical functionalities of the system based on their potential impact and likelihood of occurrence. These functionalities are the minimum operating functions of the system and are essential for its proper functioning.

Due to time constraints, the risk matrix analysis be omitted from process. However, it is recommended to evaluate the top 5 critical functionalities to ensure the system's proper operation and to mitigate potential risks.

**Objectives**

* To ensure that the application meets its intended purpose and provides a good user experience.
* To verify that all features of the application work as intended.
* To identify and report any defects or issues in the application.

**Test Strategies:**

Automated testing: This approach involves using automated tools to test the application, which can save time and effort while also providing more accurate results.

**Test Methodologies:**

The test methodologies for the BuggyCars application are Black-box testing is testing the application without any knowledge of its.

**Test Cases:**

The test cases for the BuggyCars application will be developed based on the requirements and will cover all aspects of the application, including positive and negative scenarios. Each test case will include the following information:

* Test case ID
* Test case description
* Test data
* Expected result.

**Test Environment:**

The test environment for the BuggyCars application should be set up to mimic the production environment as closely as possible but, in this situation, will follow the testers environment. The following components will be included in the test environment:

* Operating system: Windows 11
* Browser: Chrome
* Development environment: VS Code
* Testing tools: Cypress

**Test Scenarios:**

User Registration:

Verify that a user can register for an account with valid information.

Verify that a user cannot register for an account with invalid information.

User Login:

Verify that a user can log in to the application with valid credentials.

Verify that a user cannot log in to the application with invalid credentials.

Commenting System (voting system):

Verify that a user cannot vote a buggy car without being logged in.

Verify that a user can leave a comment and vote a buggy car.

Verify that a user can only vote a buggy car once.

Verify that the user can view their own rating for a buggy car.

Buggy Car Ranking System:

Verify that the buggy cars are ranked based on the total number of votes received.

Verify that the buggy cars with the highest number of votes are ranked at the top.

Verify that the ranking order is updated dynamically as new votes are received.

Buggy Car Navigation Functionality:

Verify that the user can navigate to the right page by clicking on the correct link.

Verify that the user can navigate back to the previous page by clicking on any link or button that takes the user to a different page and then clicking on the browser back button.

Verify that the buggy cars are displayed correctly.

**Test Data:**

To ensure comprehensive testing of the BuggyCars application, various types of test data will be used, including:

* Valid user registration data, including unique Login and First name, Last name Password, to test the registration functionality.
* Invalid user registration data, including duplicate Login and First name, Last name Password, to test the validation of user input.
* Valid user login credentials, including registered Login and passwords, to test the login functionality.
* Invalid user login credentials, including incorrect Login and passwords, to test the validation of user input.
* Various buggy car models, images, and descriptions, to test the browsing and rating functionality.
* Valid and invalid comments, including different lengths and special characters, to test the commenting functionality.
* Test data for edge cases, including maximum and minimum input values, to test the application's behavior under extreme conditions.

**Test Case**

**Test case ID:** **BC-REG-001**

**Test Scenario:** Verify that a user can register for an account with valid information.

**Test Case:** User registrations with valid information.

**Per-condition:** The user is on the BuggyCars registration page.

**Test Steps:**

1. Enter valid Login, first name, last name, password and confirm password in the registration form.
2. Click on the “Register” button.

**Test Data:**

* Login: Tester(time)@a
* First Name: Test
* Last Name: Er
* Password: Tester@123
* Confirm Password: Tester@123

**Expected Result:**

* A message should be displayed on the page saying, " Registration is successful."

**Actual Result:**

The user was successfully registered. A message was displayed on the page saying "Registration Successful. Please Login." The registration process passed successfully.

**Status: Pass / Fail**

**Test case ID: BC-REG-002**

**Test Scenario:** Verify that the user can not register with invalid information.

**Test Case:** The user attempts to register with invalid information.

**Per-condition:** The user is on the BuggyCars registration page.

**Test Steps:**

1. Enter “Tester@a” in the “Login” field.
2. Enter "Test" in the "First Name" field.
3. Enter "Er" in the "Last Name" field.
4. Enter "Tester" in the "Password" field.
5. Enter "Tester" in the "Confirm Password" field.
6. Click on the "Register" button.

**Test Data:**

* Login: Tester@a
* First Name: Test
* Last Name: Er
* Password: Tester
* Confirm Password: Tester

**Expected Result:**

* The registration should fail.
* An error message should be displayed indicating that the password did not conform with policy.

**Actual Result:**

* The registration failed as expected.
* An error message was displayed indicating that the password did not conform with policy: "Password not long enough".
* However, there is a suggestion for improvement to display a more user-friendly error message instead of showing the technical exception message "InvalidPasswordException".

**Status: Pass / Fail**

**Comments:** consider strike a balance between having enough granularity to catch defects and maintainability, reusability, and efficiency. This test case will not break into many small tests case. (e.g., blank input in each, invalid input in each, password does not match, duplicate login etc.)

**Test case ID: BC-LOG-001**

**Test Scenario:** Verify that a user can log in to the application with valid credentials.

**Test Case:** User login with valid credentials

**Per-condition:** The user can see login from.

**Test Steps:**

1. Enter “Tester@a” in the “Login” field.
2. Enter "Tester@123" in the "Password" field.
3. Click on the "Login" button.

**Test Data:**

* Login: Tester@a
* Password: Tester@123

**Expected Result:**

* Profile link should be visible.
* Logout should be visible.
* User can see a welcome message with their username.

**Actual Result:**

* Profile link is visible.
* Logout is visible.
* User can see a welcome message with their username.

**Status: Pass / Fail**

**Test case ID: BC-LOG-002**

**Test Scenario:** Verify that a user cannot log in to the application with invalid credentials.

**Test Case:** User login with invalid credentials

**Per-condition:** The user can see login from.

**Test Steps:**

1. Enter “Tester” in the “Login” field.
2. Enter "Tester@123" in the "Password" field.
3. Click on the "Login" button.

**Test Data:**

* Login: Tester
* Password: Tester@123

**Expected Result:** An error message indicating that the login credentials are incorrect is displayed.

**Actual Result:**

An error message “Invalid username/password” indicating that the login credentials are incorrect is displayed.

**Status: Pass / Fail**

**Test case ID: BC-VOT-001**

**Test Scenario:** Verify that a user cannot vote a buggy car without being logged in.

**Test Case:** Use try to vote without logging in.

**Per-condition:** The user is on the BuggyCars website and is not logged in.

**Test Steps:**

1. Click on the any buggy car.
2. Try to vote without logging in.

**Test Data:**

**Expected Result:**

* The user should not be able to submit a vote without logging in.
* The system should prompt the user to log in or register for an account.

**Actual Result:**

* The user was unable to submit a vote without logging in.
* The system prompted the user to log in or register for an account.

**Status: Pass / Fail**

**Test case ID: BC-VOT-002**

**Test Scenario:** Verify that a user can leave a comment and vote a buggy car.

**Test Case:**  User leave a comment and vote a buggy car.

**Per-condition:** The user is on the BuggyCars website and is logged in.

**Test Steps:**

1. Click on the any buggy car.
2. Leave a comment.
3. Click vote button.

**Test Data:**

**Comment:** comment for a buggy@123car.

**Expected Result:**

* + The system should save the comment for the selected buggy car.
  + The comment and vote should be displayed correctly on the buggy car details page.

**Actual Result:**

* The user was able to leave a comment and vote for the selected buggy car.
* The comment and vote were displayed correctly on the buggy car details page.

**Status: Pass / Fail**

**Test case ID: BC-VOT-003**

**Test Scenario:** Verify that a user can only vote a buggy car once.

**Test Case:** User try to vote a buggy car multiple time.

**Per-condition:** The user is on the BuggyCars website and is logged in.

**Test Steps:**

1. Click on the any buggy car.
2. Leave a comment and click vote button.
3. Try to vote the same buggy car again.

**Test Data:**

**Expected Result:**

* The system should not allow the user to vote the same buggy car again.
* A message should be displayed indicating that the user has already voted for the buggy car.

**Actual Result:**

The system correctly prevents the user from voting for the same buggy car again and displays a message indicating that the user has already voted for the car.

**Status: Pass / Fail**

**Test case ID: BC-VOT-004**

**Test Scenario:** Verify that the user can view their own rating for a buggy car.

**Test Case:** User view their own comment for a buggy car.

**Per-condition:** The user is on the BuggyCars website and is logged in.

**Test Steps:**

1. Click on the any buggy car.
2. Leave a comment and click vote button.
3. Reload page.
4. Check the comment for the buggy car is displayed correctly.

**Test Data:**

**Expected Result:**

* The system should save the users comment for the selected buggy car.
* The comment should be displayed correctly on the buggy car detail page.

**Actual Result:**

The system successfully saved the user's comment for the selected buggy car, and the comment was displayed correctly on the buggy car detail page after reloading the page.

**Status: Partially Pass**

**Comment:**

During the testing, the system was able to save the user's comment for the selected buggy car, and the comment was displayed correctly on the buggy car detail page after reloading the page. However, the author's name was not displayed correctly as the author row was blank. Hence, the test case is partially passed. Further investigation is required to ensure the author's name is displayed correctly.

**Test case ID: BC-RNK-001**

**Test Scenario:** Verify that the buggy cars are ranked based on the total number of votes received.

**Test Case:** Buggy cars ranked based on total number of votes.

**Per-condition:** The BuggyCars website has multiple buggy cars with votes.

**Test Steps:**

1. Verify that the buggy cars are ranked based on the total number of votes received.
2. Check that the buggy cars with the highest number of votes are ranked at the top.
3. Add a new vote to a buggy car.
4. Verify that the rank order is updates dynamically as new votes are received.

**Test Data:**

* Multiple buggy cars with votes
* New vote for a buggy car

**Expected Result:**

* The buggy cars should be ranked in descending order based on the total number of votes received.
* The buggy cars with the highest number of votes should be ranked at the top.
* The buggy car’s total number of votes should increase by one.
* The ranking order should be updated dynamically and the buggy car's new position in the ranking should reflect the change in its total number of votes.

**Actual Result:**

* The buggy cars are ranked in descending order based on the total number of votes received.
* The buggy cars with the highest number of votes are ranked at the top.
* The total number of votes for the buggy car that received a new vote has increased by one, and the ranking order has been updated dynamically to reflect the change in its total number of votes.

**Status: Pass / Fail**

**Comment:**

This test case is not repeatable due to system limitations that restrict users from voting multiple times or canceling their vote. Additionally, the voting process is irreversible, and once a vote is cast, it cannot be undone. As a result, this test case can only be executed once per user account.

**Test case ID: BC-NAV-001**

**Test Scenario:** Verify that the user can navigate to the BuggyCars home page.

**Test Case:** User navigates to the BuggyCars home page.

**Per-condition:** The user is on any page of the BuggyCars website.

**Test Steps:**

1. Click on the BuggyCars logo located on the top left corner of the page.

**Test Data:**

**Expected Result:**

* The user should be redirected to the BuggyCars home page.

**Actual Result:**

The test case has been broken down into four sub-test cases, and all of them pass except for "BC-NAV-001-2" which is for the popular car link. The link on the page is incorrect, it does not redirect the user to the homepage. The other sub-test cases pass successfully, and the user is redirected to the homepage.

**Status: Partially Pass**

**Comment:**

The test case has been broken down into sub-test cases due to the four different types of pages on the website. However, there is an issue with the popular make link, as it does not redirect the user to the homepage. This issue needs to be resolved to ensure that the user can navigate to the homepage from any page on the website.

**Test case ID: BC-NAV-002**

**Test Scenario:** Verify that the user can navigate to the right page by clicking on the correct link.

**Test Case:** User navigates to the BuggyCars overall ranking page.

**Per-condition:** The user is on the home page of BuggyCars website.

**Test Steps:**

1. Click on the "Overall Ranking" link.

**Test Data:**

**Expected Result:**

* The user should be redirected to the BuggyCars overall ranking page.

**Actual Result:**

* The user was able to navigate to the "Overall Rating" page by clicking the "Overall Rating" link.

**Status: Pass / Fail**

**Test case ID: BC-NAV-003**

**Test Scenario:** Verify that the user can navigate to the right page by clicking on the correct link.

**Test Case:** User navigates to the BuggyCars popular make page.

**Per-condition:** The user is on the home page of BuggyCars website.

**Test Steps:**

1. Click on the "Popular Make" link.

**Test Data:**

**Expected Result:**

* The user should be redirected to the BuggyCars popular make page.

**Actual Result:**

* The user was able to navigate to the "Popular Make" page by clicking the "Popular Make" link.

**Status: Pass / Fail**

**Test case ID: BC-NAV-004**

**Test Scenario:** Verify that the user can navigate to the right page by clicking on the correct link.

**Test Case:** User navigates to the BuggyCars car detail page.

**Per-condition:** The user is on the home page of BuggyCars website.

**Test Steps:**

1. Click on the "Popular Model" link.

**Test Data:**

**Expected Result:**

* The user should be redirected to the BuggyCars car detail page for the selected buggy car model.

**Actual Result:**

* The user was able to navigate to the "Popular Car" page by clicking the "Popular Car" link.

**Status: Pass / Fail**

**Test case ID: BC-NAV-005**

**Test Scenario:** Verify that the user can navigate back to the previous page.

**Test Case:** User navigates back to the previous page.

**Per-condition:** The user is on any page of the BuggyCars website.

**Test Steps:**

1. Click on any link or button that takes the user to a different page.
2. Click on the browser back button.

**Test Data:**

**Expected Result:**

* The user should be redirected back to the previous page.

**Actual Result:**

* The user is successfully navigated back to the previous page.

**Status: Pass / Fail**

**Test case ID: BC-NAV-006**

**Test Scenario:** Verify that the buggy cars are displayed correctly.

**Test Case:** User views the buggy cars.

**Per-condition:** The user is on the BuggyCars Overall Ranking page or the BuggyCars popular make page.

**Test Steps:**

1. Check that the buggy cars are displayed in a list format with the image, make, model, rank, vote, and comment.
2. Check that each buggy car model is represented by a link to its detail page with vote page.

**Test Data:**

**Expected Result:**

* The buggy cars should be displayed correctly, with each car represented by a thumbnail image, the car model, and the overall rating.

**Actual Result:**

* The buggy cars were displayed correctly in both the popular make page and overall ranking page. Each car was represented by a thumbnail image, the car model, and the overall rating.
* The list format included the image, make, model, rank, vote, and comment.
* Each buggy car model was represented by a link to its detail page with the vote page.

**Status: Pass / Fail**