# Asher Khalid QA Test Task – EliteIT asherkhalid@hotmail.com

# **Introduction**

The purpose of this test plan is to outline the testing approach, objectives, scope, environment, and schedule for testing the online registration form of the website. Users can create an account by entering their details on the registration form, which is an essential component of the website.

## **Objectives**

The objectives of testing the registration form are:

1. To validate that the form records user data accurately.
2. To validate the form input fields according to specified requirements.
3. To verify that appropriate error messages are displayed for invalid inputs.
4. To verify that, after being filled out correctly, the form properly submits the data to the server.

## **Scope**

The scope of this test plan includes:

1. **Input Validation:** Functional testing of the registration form's input fields.
2. **Data Validation:** Validation of data entered into the form fields.
3. **Usability:** Test the form's usability to make sure users have a positive experience.
4. **Compatibility:** Compatibility testing across different browsers and devices.

## **Test Environment**

The test environment will include:

* **Browsers:** Google Chrome, Firefox, Safari, Microsoft Edge
* **Operating Systems:** Windows, macOS, Linux
* **Devices:** Desktop, Tablet, Mobile

## **Test Data**

Test data will include:

* Valid and invalid data entered into the input fields.
* Different combinations of Newsletter Subscription options.

## **Test Scenarios**

Test scenarios will cover:

* Positive scenarios: Valid input data.
* Negative scenarios: Invalid or missing input data.
* Cross-browser and cross-device testing.
* Boundary testing for input fields.

## **Test Execution Schedule**

* **Test planning and preparation:** 1 week
* **Test case design:** 1 week
* **Test environment setup:** 1 week
* **Test execution:** 2 weeks
* **Bug reporting and retesting:** Ongoing during test execution

## **Risks** **and** **Assumptions**

**Risks:**

1. **Inconsistent Behaviour:** Inconsistent behaviour across different browsers and devices.
2. **Design Changes:** Last-minute changes to form design or requirements.
3. **Integration Issues:** Potential integration issues with third-party services or APIs.

**Assumptions:**

1. **Functional Implementation:** All required functionalities are implemented.
2. **Stable API:** The server-side API for form submission is stable and available.
3. **Test Data Availability:** Test data is available and representative of real-world scenarios.

## **Test Cases for Registration Form**

### **Test Case 1: Valid Registration**

* **Test Case ID:** T001
* **Test Case Description:** Verify that a user can successfully register with valid input data.
* **Preconditions:** The registration form is accessible.
* **Test Steps:**
  1. Enter a valid Full Name.
  2. Enter a valid Email Address.
  3. Enter a valid Password.
  4. Enter the same Password in the Confirm Password field.
  5. Select a valid Date of Birth.
  6. Select a Gender.
  7. Select "Yes" or "No" for Newsletter Subscription.
  8. Click the Submit Button.
* **Expected Results:** The form submits successfully, and the user receives a confirmation message.
* **Post-conditions:** The user's data is saved in the database.

### **Test Case 2: Invalid Email Address**

* **Test Case ID:** T002
* **Test Case Description:** Verify that the form displays an error message for an invalid email address.
* **Preconditions:** The registration form is accessible.
* **Test Steps:**
  1. Enter a valid Full Name.
  2. Enter an invalid Email Address.
  3. Enter a valid Password.
  4. Enter the same Password in the Confirm Password field.
  5. Select a valid Date of Birth.
  6. Select a Gender.
  7. Select "Yes" or "No" for Newsletter Subscription.
  8. Click the Submit Button.
* **Expected Results:** An error message is displayed indicating the email address is invalid.
* **Post-conditions:** The form is not submitted, and the user can correct the email address.

### **Test Case 3: Password Mismatch**

* **Test Case ID:** T003
* **Test Case Description:** Verify that the form displays an error message when the password and confirm password do not match.
* **Preconditions:** The registration form is accessible.
* **Test Steps:**
  1. Enter a valid Full Name.
  2. Enter a valid Email Address.
  3. Enter a valid Password.
  4. Enter a different Password in the Confirm Password field.
  5. Select a valid Date of Birth.
  6. Select a Gender.
  7. Select "Yes" or "No" for Newsletter Subscription.
  8. Click the Submit Button.
* **Expected Results:** An error message is displayed indicating the passwords do not match.
* **Post-conditions:** The form is not submitted, and the user can correct the passwords.

## **Scenario** **Identification**

### **Positive** **Scenarios**

1. **Valid Registration:** A user enters valid data in all fields and successfully submits the form.
   * **Steps:** Fill in all fields with valid data and click Submit.
   * **Expected Outcome:** Form submits successfully, data stored successfully in database, and user receives a confirmation message.

#### **Valid Registration with Different Gender Selection**:

* + **Steps:** Fill in all fields with valid data, select "Other" for Gender, and click Submit.
  + **Expected Outcome:** Form submits successfully, data stored successfully in database, and user receives a confirmation message.

### **Negative** **Scenarios**

1. **Missing Required Fields:**
   * **Steps:** Attempt to submit the form with one or more required fields left blank (e.g., Full Name or Email Address).
   * **Expected Outcome:** The form should not submit, and appropriate error messages should be displayed for the missing fields.
2. **Invalid Date of Birth Format:**
   * **Steps:** Enter an invalid Date of Birth (e.g., "31-02-2000") and attempt to submit the form.
   * **Expected Outcome:** The form should not submit, and an error message should be displayed indicating an invalid date format.

## **Positive Scenario Test Scripts**

### **Scenario: Valid Registration**

1. from selenium import webdriver
2. from selenium.webdriver.common.by import By
3. from selenium.webdriver.common.keys import Keys
4. from selenium.webdriver.support.ui import Select
5. import time
6. # Set up the WebDriver
7. driver = webdriver.Chrome(executable\_path='C:/Users/asher/Downloads/chromedriver-win64/chromedriver.exe')
8. def test\_valid\_registration():
9. # Navigate to the registration page
10. driver.get('http://websitetest.com/register')
11. # Fill in the form fields
12. driver.find\_element(By.NAME, 'fullName').send\_keys('Asher Khalid')
13. driver.find\_element(By.NAME,'email').send\_keys('asherkhalid@hotmail.com')
14. driver.find\_element(By.NAME,'password').send\_keys('$Password12345')
15. driver.find\_element(By.NAME,'confirmPassword').send\_keys('$Password12345')
16. driver.find\_element(By.NAME,'dateOfBirth').send\_keys('27-07-2002')
17. Select(driver.find\_element(By.NAME,'gender')).select\_by\_visible\_text('Male')
18. driver.find\_element(By.NAME,'newsletter\_yes').click()
19. # Submit the form
20. driver.find\_element(By.CSS\_SELECTOR,'button[type="submit"]').click()
21. # Verify the success message
22. time.sleep(3)  # Wait for the success message to appear
23. success\_message = driver.find\_element(By.XPATH,'//\*[contains(text(), "Registration successful")]')
24. assert success\_message.is\_displayed(), "Success message is not displayed"
25. def teardown():
26. # Close the browser
27. driver.quit()
28. if \_\_name\_\_ == "\_\_main\_\_":
29. try:
30. test\_valid\_registration()
31. finally:
32. teardown()

### **Scenario: Valid Registration with Different Gender Selection**

1. from selenium import webdriver
2. from selenium.webdriver.common.by import By
3. from selenium.webdriver.common.keys import Keys
4. from selenium.webdriver.support.ui import Select
5. import time
6. # Set up the WebDriver
7. driver = webdriver.Chrome(executable\_path='C:/Users/asher/Downloads/chromedriver-win64/chromedriver.exe')
8. def test\_valid\_registration():
9. # Navigate to the registration page
10. driver.get('http://websitetest.com/register')
11. # Fill in the form fields
12. driver.find\_element(By.NAME, 'fullName').send\_keys('Asher Khalid')
13. driver.find\_element(By.NAME,'email').send\_keys('asherkhalid@hotmail.com')
14. driver.find\_element(By.NAME,'password').send\_keys('$Password12345')
15. driver.find\_element(By.NAME,'confirmPassword').send\_keys('$Password12345')
16. driver.find\_element(By.NAME,'dateOfBirth').send\_keys('27-07-2002')
17. Select(driver.find\_element(By.NAME,'gender')).select\_by\_visible\_text('Other')
18. driver.find\_element(By.NAME,'newsletter\_yes').click()
19. # Submit the form
20. driver.find\_element(By.CSS\_SELECTOR,'button[type="submit"]').click()
21. # Verify the success message
22. time.sleep(3)  # Wait for the success message to appear
23. success\_message = driver.find\_element(By.XPATH,'//\*[contains(text(), "Registration successful")]')
24. assert success\_message.is\_displayed(), "Success message is not displayed"
25. def teardown():
26. # Close the browser
27. driver.quit()
28. if \_\_name\_\_ == "\_\_main\_\_":
29. try:
30. test\_valid\_registration()
31. finally:
32. teardown()

### 