Test Task: Documentation and Test Scenario Identification

**Introduction**

The purpose of this test plan is to outline the strategy and approach for testing the online registration form for a website. The form collects user information and must validate inputs, display appropriate error messages, and successfully submit data upon correct completion.

**Objectives**

* Ensure that the registration form validates user inputs correctly.
* Verify that appropriate error messages are displayed for invalid user inputs.
* Confirm that the form successfully submits data to the server upon correct completion.
* Validate the user experience and form functionality.

**Scope**

This test plan covers the functional testing of the online registration form, including validation checks, error messages, and successful form submission. It does not include non-functional testing.

**Test Environment**

* Browser: Latest versions of Chrome, Firefox, Safari, and Edge.
* Operating Systems: Windows 10, macOS, Linux.
* Test Server: Staging environment with the latest version of the website deployed.

**Test Data**

* Valid and invalid user names, email addresses, and passwords.
* Different combinations of gender and newsletter subscription choices.
* Various date of birth entries.

**Test Scenarios**

1. Valid form submission (happy flow).

2. Invalid email format.

3. Passwords do not match.

4. Password length less than 8 characters.

5. Missing mandatory fields.

**Test Execution Schedule**

* Test planning: 1 days
* Test case design: 2 days
* Test case execution: 3 days
* Defect logging and retesting: 2 days
* Test summary report: 1 day

**Risks and Assumptions**

* The test environment will be stable and available throughout the testing period.
* Test data will be accurate and accessible.
* The registration form functionality will not undergo significant changes during testing.
* Cross-browser and cross-platform compatibility is assumed to be already addressed.

**Test cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Preconditions** | **Test Steps** | **Expected Results** | **Post-conditions** |
| TC01 | Valid form submission | The user is navigated to the Registration form | 1. Open the registration form. 2. Enter a valid full name. 3. Enter a valid email address. 4. Enter a valid password. 5. Confirm password. 6. Select date of birth. 7. Select gender. 8. Choose a newsletter subscription. 9. Click submit. | Form submits successfully and navigates to the welcome page. | The user is registered Successfully. |
| TC02 | Invalid email format | The user is navigated to the Registration form | 1. Open the registration form. 2. Enter a valid full name. 3. Enter an invalid email address. 4. Enter a valid password. 5. Confirm password. 6. Select date of birth. 7. Select gender. 8. Choose a newsletter subscription. 9. Click submit. | Error message "Invalid email address" is displayed. | The user is prompted to correct the email. |
| TC03 | Passwords do not match | The user is navigated to the Registration form | 1. Open the registration form. 2. Enter a valid full name. 3. Enter a valid email address. 4. Enter a valid password. 5. Enter a different password in the confirm password. 6. Select date of birth. 7. Select gender. 8. Choose a newsletter subscription. 9. Click submit. | Error message "Passwords do not match" is displayed. | The user is prompted to match passwords. |

**Scenario Identification**

**Positive Scenarios**

1. **Valid Form Submission:**

* + **Steps:** User fills in all fields with valid data and clicks submit.
  + **Expected Outcome:** The form is submitted successfully, and the user is navigated to the next page.

**2. Newsletter Subscription:**

* + **Steps:** User fills in all fields with valid data, opts for newsletter subscription, and clicks submit.
  + **Expected Outcome:** The form is submitted successfully, and the user is added to the newsletter subscription list.

**Negative Scenarios**

1. **Invalid Email Format:**

* + **Steps:** The user fills in all fields but enters an invalid email format (e.g. [hussain@.com](mailto:hussain@.com)) and clicks submit.
  + **Expected Outcome:** An error message "Invalid email address" is displayed.

**2. Password Length Less Than 8 Characters:**

* + **Steps:** User fills in all fields but enters a password with less than 8 characters and clicks submit.
  + **Expected Outcome:** An error message "Password must be at least 8 characters long" is displayed.

**Automated Test Scripts**

* **Positive Scenario: Valid Form Submission**

|  |
| --- |
| from selenium import webdriver from selenium.webdriver.common.by import By from selenium.webdriver.support.ui import WebDriverWait from selenium.webdriver.support import expected\_conditions as EC from selenium.webdriver.common.keys import Keysdef test\_valid\_form\_submission():     driver = webdriver.Chrome()     driver.get("<https://example.com/register>")    # Fill in the form     driver.find\_element(By.ID, "full\_Name").send\_keys("Hussain")     driver.find\_element(By.ID, "email").send\_keys("[hussain@yopmail.com](mailto:hussain@yopmail.com)")     driver.find\_element(By.ID, "password").send\_keys("Hussain@123")     driver.find\_element(By.ID, "confirm\_Password").send\_keys("Hussain@123")     driver.find\_element(By.ID, "dob").send\_keys("1990-01-01")     driver.find\_element(By.ID, "gender").send\_keys("Male")     driver.find\_element(By.ID, "newsletter").click()     driver.find\_element(By.ID, "submit").click()      # Wait for the welcome page to load and check for the welcome message      WebDriverWait(driver, 10).until(EC.url\_contains("/welcome"))     assert "Welcome, Hussain" in driver.page\_source    driver.quit()if *name* == "*main*":     test\_valid\_form\_submission() |

* **Positive Scenario: Newsletter Subscription:**

|  |
| --- |
| from selenium import webdriver from selenium.webdriver.common.by import By from selenium.webdriver.support.ui import WebDriverWait from selenium.webdriver.support import expected\_conditions as EC from selenium.webdriver.common.keys import Keys import requestsdef test\_newsletter\_subscription():     driver = webdriver.Chrome()     driver.get("<https://example.com/register>")    # Fill in the form     driver.find\_element(By.ID, "full\_Name").send\_keys("Hussain")     driver.find\_element(By.ID, "email").send\_keys("[hussain@yopmail.com](mailto:hussain@yopmail.com)")     driver.find\_element(By.ID, "password").send\_keys("Hussain@123")     driver.find\_element(By.ID, "confirm\_Password").send\_keys("Hussain@123")     driver.find\_element(By.ID, "dob").send\_keys("1990-01-01")     driver.find\_element(By.ID, "gender").send\_keys("Male")     driver.find\_element(By.ID, "newsletter").click()     driver.find\_element(By.ID, "submit").click()       # Wait for the welcome page to load and check for the welcome message      WebDriverWait(driver, 10).until(EC.url\_contains("/welcome"))     assert "Welcome, Hussain" in driver.page\_source      # Check if the user is subscribed to the newsletter      response = requests.get("<https://example.com/api/newsletter-subscribers>")     assert "[hussain@yopmail.com.com](mailto:hussain@yopmail.com.com)" in response.text    driver.quit()if \_*name*\_ == "\_*main*\_":     test\_newsletter\_subscription() |