2024

Test Plan Document

Test plan outline, test cases, scenario identification, and test scripts

SANA ARIF

ELITE IT | London, England

Table of Contents

1 Test Plan for Online Registration Form

1.1 Introduction

This test plan outlines the strategy and approach for testing the online registration form for the website. The goal is to ensure that the form functions correctly, validates inputs accurately, and submits data successfully.

1.2 Objectives

- Verify all input fields and options are present.
- Ensure validation rules for each input field work as expected.
- Check that error messages are displayed correctly for invalid inputs.
- Confirm the form is submitted successfully with valid input.
- Verify data is correctly sent to the server upon form submission.
- Ensure the user interface is user-friendly and accessible.

1.3 Scope

- Functional testing of all input fields and form controls.
- Validation testing for input constraints (e.g., password length, email format).
- Usability testing to ensure a smooth user experience.
- Compatibility testing across different browsers and devices.
- Security testing to ensure data is protected during submission.

1.4 Test Environment

Let's assume that we have the following environment for the form.

1.4.1 Operating Systems

- Windows, macOS, Linux, Android, iOS

1.4.2 Browsers

- Chrome, Firefox, Safari, Edge, Internet Explorer

1.4.3 Devices

- Desktop, Laptop, Tablet, Smartphone

1.4.4 Network Conditions

- Different bandwidths (high speed, moderate speed, low speed)

1.5 Test Data

- Full Name: Various names including special characters, long names, and short names.
- Email Address: Valid and invalid email formats.
- Password: Valid passwords (at least 8 characters), invalid passwords (less than 8 characters, without special characters).
- Confirm Password: Matching and non-matching passwords.
- Date of Birth: Valid dates, invalid dates, future dates, edge cases like leap years.
- Gender: Selection of all options (Male, Female, Other).
- Newsletter Subscription: Both options (Yes, No).

1.6 Test Scenarios

1.6.1 Presence of Elements

- Verify that all input fields and the submit button are present.
- Verify that the gender selection and newsletter subscription options are available.

1.6.2 Field Validation

- Test input of valid and invalid email addresses.
- Test password length requirement (minimum 8 characters).
- Test confirm password field for matching passwords.
- Test valid and invalid date of birth entries.
- Test mandatory fields (Full Name, Email Address, Password, Confirm Password).

1.6.3 Error Messages

- Verify that appropriate error messages are displayed for invalid inputs.
- Verify that error messages disappear when corrected input is provided.

1.6.4 Form Submission

- Test successful form submission with valid inputs.
- Test server response and data submission.
- Test behavior with empty fields.

1.6.5 Usability

- Test the ease of navigation and user experience.
- Verify that form fields are accessible, and labels are clear.

1.6.6 Compatibility

- Test form functionality across different browsers and devices.

1.6.7 Security

- Verify that password fields mask input.
- Test secure data transmission during form submission.

1.7 Test Execution Schedule

1.7.1 Week 1

- Set up the test environment.
- Prepare test data.
- Conduct initial testing for the presence of elements and field validation.

1.7.2 Week 2

- Perform validation testing.
- Test error messages and form submission.

1.7.3 Week 3

- Conduct usability testing.
- Perform compatibility testing across different browsers and devices.

1.7.4 Week 4

- Conduct security testing.
- Review and document test results.

- Report defects and follow up on resolutions.

1.7.5 Risks and Assumptions

Risks

- Potential delays in test environment setup.
- Unavailability of required devices or browsers for compatibility testing.
- Changes in form design or validation rules during the testing phase.

Assumptions

- The form design and validation rules are finalized and stable.
- Test environment and necessary resources will be available as planned.
- No major changes to the website that would affect the registration form during the testing period.

2 Test Cases

2.1 Test Case 1: Validate Email Field

Test Case ID: TC_RF_001

Test Case Description: Verify that the email field accepts only valid email addresses.

Preconditions: The user is on the registration form page.

Test Steps:

1. Enter "invalidemail" in the Email Address field.

2. Click the Submit button.

3. Observe the validation message.

4. Enter "user@domain" in the Email Address field.

5. Click the Submit button.

6. Observe the validation message.

7. Enter "user@domain.com" in the Email Address field.

8. Click the Submit button.

9. Observe the validation message.

Expected Results:

1. An error message "Please enter a valid email address." is displayed.

2. An error message "Please enter a valid email address." is displayed.

3. The error message disappears, and the form accepts the input.

Post-conditions: The Email field accepts valid email formats and rejects invalid ones.

2.2 Test Case 2: Password and Confirm Password Match

Test Case ID: TC_RF_002

Test Case Description: Verify that the password and confirm password fields match.

Preconditions: The user is on the registration form page.

Test Steps

1. Enter "ValidPassword1!" in the Password field.

2. Enter "InvalidPassword2@" in the Confirm Password field.

3. Click the Submit button.

4. Observe the validation message.

5. Enter "ValidPassword1!" in both the Password and Confirm Password fields.

6. Click the Submit button.

7. Observe the validation message.

Expected Results

1. An error message "Passwords do not match." is displayed.

2. The error message disappears, and the form accepts the input.

Post-conditions: Password and Confirm Password fields must match for successful submission.

2.3 Test Case 3: Date of Birth Validation

Test Case ID: TC_RF_003

Test Case Description: Verify that the Date of Birth field does not accept future dates.

Preconditions: The user is on the registration form page.

Test Steps:

1. Enter "12/31/2025" in the Date of Birth field.

2. Click the Submit button.

3. Observe the validation message.

4. Enter "12/31/2000" in the Date of Birth field.

5. Click the Submit button.

6. Observe the validation message.

Expected Results:

1. An error message "Please enter a valid date of birth." is displayed.

2. The error message disappears, and the form accepts the input.

Post-conditions: The Date of Birth field does not accept future dates and only accepts valid past dates.

3 Scenario Identification

3.1 Positive Scenarios

3.1.1 Scenario 1: Successful Registration

Steps:

1. Open the registration form page.

2. Enter a valid Full Name (e.g., "John Doe").

3. Enter a valid Email Address (e.g., "john.doe@example.com").

4. Enter a valid Password (e.g., "Password123!").

5. Enter the same password in the Confirm Password field.

6. Enter a valid Date of Birth (e.g., "01/01/1990").

7. Select a Gender option (e.g., "Male").

8. Choose "Yes" or "No" for Newsletter Subscription.

9. Click the Submit button.

Expected Outcome: The form is submits successfully, and a confirmation message appears indicating that registration was successful. The data is correctly sent to the server.

3.2 Scenario 2: Opt-Out of Newsletter Subscription

Steps:

1. Open the registration form page.

2. Enter a valid Full Name (e.g., "Jane Smith").

3. Enter a valid Email Address (e.g., "jane.smith@example.com").

4. Enter a valid Password (e.g., "SecurePass123!").

5. Enter the same password in the Confirm Password field.

6. Enter a valid Date of Birth (e.g., "02/14/1985").

7. Select a Gender option (e.g., "Female").

8. Choose "No" for Newsletter Subscription.

9. Click the Submit button.

Expected Outcome: The form submits successfully, and a confirmation message appears indicating that registration was successful. The user is not subscribed to the newsletter.

3.3 Negative Scenarios

3.3.1 Scenario 1: Invalid Email Format

Steps:

1. Open the registration form page.

2. Enter a valid Full Name (e.g., "John Doe").

3. Enter an invalid Email Address (e.g., "john.doe.com").

4. Enter a valid Password (e.g., "Password123!").

5. Enter the same password in the Confirm Password field.

6. Enter a valid Date of Birth (e.g., "01/01/1990").

7. Select a Gender option (e.g., "Male").

8. Choose "Yes" or "No" for Newsletter Subscription.

9. Click the Submit button.

Expected Outcome: An error message appears indicating that the email address is invalid. The form does not submit, and the user is prompted to correct the email address.

3.3.2 Scenario 2: Passwords Do Not Match

Steps:

1. Open the registration form page.

2. Enter a valid Full Name (e.g., "Jane Smith").

3. Enter a valid Email Address (e.g., "jane.smith@example.com").

4. Enter a valid Password (e.g., "SecurePass123!").

5. Enter a different password in the Confirm Password field (e.g., "SecurePass456!").

6. Enter a valid Date of Birth (e.g., "02/14/1985").

7. Select a Gender option (e.g., "Female").

8. Choose "Yes" or "No" for Newsletter Subscription.

9. Click the Submit button.

Expected Outcome: An error message appears indicating that the passwords do not match. The form does not submit, and the user is prompted to correct the passwords.

4 Automated Test Scripts

Create a new test file under `cypress/integration` (e.g., `registration_form.spec.js`) and add the following scripts:

4.1 JavaScript

```
/// <reference types="cypress" />

describe('Registration Form', () => {
  const validFullName = 'John Doe';
  const validEmail = 'john.doe@example.com';
  const validPassword = 'Password123!';
  const validDateOfBirth = '1990-01-01'; // YYYY-MM-DD format

  beforeEach(() => {
    cy.visit('/registration'); // Adjust the URL according to your application
  });

  it('should successfully register with valid inputs', () => {
    cy.get('input[name="fullName"]').type(validFullName);
    cy.get('input[name="email"]').type(validEmail);
    cy.get('input[name="password"]').type(validPassword);
    cy.get('input[name="confirmPassword"]').type(validPassword);
    cy.get('input[name="dob"]').type(validDateOfBirth);
    cy.get('input[name="gender"][value="Male"]').check();
    cy.get('input[name="newsletter"][value="Yes"]').check();
    cy.get('button[type="submit"]').click();

    cy.get('.success-message').should('contain', 'Registration successful');
  });

  it('should successfully register without subscribing to the newsletter', () => {
    cy.get('input[name="fullName"]').type('Jane Smith');
    cy.get('input[name="email"]').type('jane.smith@example.com');
    cy.get('input[name="password"]').type('SecurePass123!');
    cy.get('input[name="confirmPassword"]').type('SecurePass123!');
    cy.get('input[name="dob"]').type('1985-02-14');
    cy.get('input[name="gender"][value="Female"]').check();
    cy.get('input[name="newsletter"][value="No"]').check();
    cy.get('button[type="submit"]').click();

    cy.get('.success-message').should('contain', 'Registration successful');
  });
});
```

4.2 Explanation of the Scripts

4.2.1 Imports and Setup:

/// <reference types="cypress" /> is a directive that provides autocomplete and IntelliSense support for Cypress in your editor.

describe blocks define the test suite.

4.2.2 Common Variables:

Define common variables such as validFullName, validEmail, validPassword, and validDateOfBirth.

Before Each Test:

cy.visit('/registration') navigates to the registration page before each test.

Test Case: Successful Registration with Newsletter Subscription:

Fill in the form fields with valid data.

Select the gender and newsletter subscription options.

Click the submit button.

Verify the success message using cy.get('.success-message').should('contain', 'Registration successful').

Test Case: Successful Registration without Newsletter Subscription:

Follow similar steps as above but select the "No" option for the newsletter subscription.

Verify the success message.