NAME: Kinza Ahmed

ELITE IT TEAM

First assessment

Role: Jr. QA Engineer

QA ENGINEER assessment – Elite IT Team

Test Task: Documentation and Test Scenario Identification

Background:

Imagine you are a QA Engineer responsible for testing a simple online registration form for a website. The registration form collects the following information from users:

- **Full Name**
- **Email Address**
- **Password (at least 8 characters)**
- **Confirm Password**
- **Date of Birth**
- **Gender (Male/Female/Other)**
- **Newsletter Subscription (Yes/No)**
- **Submit Button**

The form should validate inputs, display appropriate error messages, and successfully submit the data to the server upon correct completion.

Task:

Documentation:

a. Provide a test plan outline for testing the registration form. Include the following sections:

- **Introduction:**

This test plan outlines the strategy to validate the accurate working of the online registration form. The goal is to identify errors, validate inputs and prompt user for any errors. And upon successful completion, the form should submit on the server.

- **Objectives**

- Ensure the form accepts valid input and prompts error on invalid input from user.
- All validation messages should be correct and indicate the user what to fix.
- Confirm the user when the form is successfully submitted on the server and data is transmitted.
- Ensure the form's accessibility for authorized users.

- **Scope**

- **Functional Testing**: Verify all form fields and their validation rules.

- **Usability Testing:** Ensure the form is user-friendly and provides clear error messages.
- **Performance Testing:** Check the form's responsiveness and load time.
- **Security Testing**: Ensure the form handles data securely, especially sensitive information like passwords.
- **Compatibility Testing**: Verify the form works across different browsers and devices.

- **Test Environment**

- **Browsers**: Chrome, Firefox, Safari, Edge, Internet Explorer
- **Devices:** Desktop, Tablet, Mobile
- **Operating Systems**: Windows, macOS, iOS, Android

- **Test Data**

Full name field:
- Should include full name (first name, middle name, last name), where middle name should not be marked as mandatory field.
- Should not include numbers and special characters.
- Cannot be left as an empty field.

Email address field:
Valid examples:

- "john.doe@example.com"
- "anna.maria.smith@example.co.uk"

- Should not miss @ symbol.
- Should not miss domain.
- Cannot be left as an empty field.

Password field:
- Should be a strong password consisting of 1 uppercase letter, numbers and special characters.
- Should contain at least 8 characters.
- Cannot be left as an empty field.

Confirm Password field:
- Should match the first password field.
- Cannot be left as an empty field.

Date of birth field:
- Should be in either DD-MM-YYYY or MM-DD-YYYY format.
- Should prompt user on invalid data.

Gender field:
Include three options:
- Male
- Female

- Other

User should be able to select one.

Newsletter subscription field:
Ask the user to subscribe or not. Two options as:

- Yes
- No

Select one.

Submit button:

The user should be able to submit the form after checking all fields as correct.
Otherwise highlight with red on missing or incorrect fields.

- **Test Scenarios**

1. Full Name Field

1. **TS01**: Verify the Full Name field accepts alphabetic characters.
2. **TS02**: Verify the Full Name field rejects numeric characters.
3. **TS03**: Verify the Full Name field rejects special characters.
4. **TS04**: Verify the Full Name field accepts a minimum and maximum number of characters.
5. **TS05**: Verify the Full Name field is mandatory.

2. Email Address Field

1. **TS06**: Verify the Email Address field accepts valid email formats.
2. **TS07**: Verify the Email Address field rejects invalid email formats.
3. **TS08**: Verify the Email Address field is mandatory.

3. Password Field

1. **TS09**: Verify the Password field accepts passwords with a minimum of 8 characters.
2. **TS10**: Verify the Password field accepts alphanumeric characters and special symbols.
3. **TS11**: Verify the Password field is mandatory.
4. **TS12**: Verify the Password field masks the entered characters.

4. Confirm Password Field

1. **TS13**: Verify the Confirm Password field matches the Password field.
2. **TS14**: Verify the Confirm Password field is mandatory.
3. **TS15**: Verify the Confirm Password field masks the entered characters.

5. Date of Birth Field

1. **TS16**: Verify the Date of Birth field accepts valid date formats.
2. **TS17**: Verify the Date of Birth field rejects invalid date formats.
3. **TS18**: Verify the Date of Birth field is mandatory.

6. Gender Field

1. **TS19**: Verify the Gender field allows selection from Male, Female, or Other.
2. **TS20**: Verify the Gender field is mandatory.

7. Newsletter Subscription Field

1. **TS21**: Verify the Newsletter Subscription field allows selection between Yes or No.
2. **TS22**: Verify the Newsletter Subscription field is optional.

8. Submit Button

1. **TS23**: Verify the form submits successfully when all inputs are valid.
2. **TS24**: Verify the form does not submit if any input is invalid.
3. **TS25**: Verify appropriate error messages are displayed for invalid inputs.
4. **TS26**: Verify error messages disappear once the input is corrected.

9. Usability Testing

1. **TS27**: Verify the tab order of form fields.
2. **TS28**: Verify the clarity and visibility of validation messages.
3. **TS29**: Verify the form layout and alignment.
4. **TS30**: Verify the form's instructions and labels are clear and concise.

10. Performance Testing

1. **TS31**: Verify the form loads within an acceptable time.
2. **TS32**: Verify the form's responsiveness on different devices.
3. **TS33**: Verify the form handles multiple simultaneous submissions gracefully.

11. Security Testing

1. **TS34**: Verify passwords are masked while typing.
2. **TS35**: Verify the form uses HTTPS for secure data transmission.
3. **TS36**: Verify the form is protected against SQL injection.
4. **TS37**: Verify the form is protected against cross-site scripting (XSS).

12. Compatibility Testing

1. **TS38**: Verify the form works correctly on different browsers (Chrome, Firefox, Safari, Edge, Internet Explorer).
2. **TS39**: Verify the form is responsive on different devices (desktop, tablet, mobile).
3. **TS40**: Verify the form works correctly on different operating systems (Windows, macOS, iOS, Android).

- **Test Execution Schedule**
  - Test Case Design: [Start Date] - [End Date]
  - Test Execution: [Start Date] - [End Date]
  - Bug Reporting and Fixes: [Start Date] - [End Date]
  - Final Verification: [Start Date] - [End Date]

- **Risks and Assumptions**

  - Identify potential risk such as delayed requirements, form’s response to technical disturbances or power outrages
  - Form’s accessibility on forgetting password.
  - Unauthorized access of forms.
  - How the form displays on different devices and browsers?

b. Write test cases for at least three scenarios covering different aspects of the registration form. Each test case should include:

- Test Case ID
- Test Case Description
- Preconditions
- Test Steps
- Expected Results
- Post-conditions

Test Case 1: Full Name Field - Alphabetic Characters

Test Case ID:

TC01

Test Case Description:

Verify the Full Name field accepts alphabetic characters.

Preconditions:

- User is on the registration form page.

Test Steps:

1. Locate the Full Name field.
2. Enter "John Doe" into the Full Name field.
3. Move to the next field or click out of the Full Name field.

Expected Results:

- The Full Name field accepts the input "John Doe" without displaying any error messages.

Post-conditions:

- The Full Name field contains the entered value "John Doe".

Test Case 2: Email Address Field - Valid Email Format

Test Case ID:

TC02

Test Case Description:

Verify the Email Address field accepts a valid email format.

Preconditions:

- User is on the registration form page.

Test Steps:

1. Locate the Email Address field.
2. Enter "john.doe@example.com" into the Email Address field.
3. Move to the next field or click out of the Email Address field.

Expected Results:

- The Email Address field accepts the input "john.doe@example.com" without displaying any error messages.

Post-conditions:

- The Email Address field contains the entered value "john.doe@example.com".

Test Case 3: Password Field - Minimum Length Requirement

Test Case ID:

TC03

Test Case Description:

Verify the Password field rejects passwords with less than 8 characters.

Preconditions:

- User is on the registration form page.

Test Steps:

1. Locate the Password field.
2. Enter "pass" (4 characters) into the Password field.

3. Move to the next field or click out of the Password field.

Expected Results:

- The Password field displays an error message indicating that the password must be at least 8 characters long.
- The form should not be allowed to submit if this field is invalid.

Post-conditions:

- The Password field contains the entered value "pass" and displays an error message.

Scenario Identification:

a. Identify at least two positive and two negative scenarios for the registration form. For each scenario, briefly describe the steps a user would take and the expected outcome.

Positive Scenario 1: Successful Registration with Valid Data

Steps:

1. User navigates to the registration form page.
2. User enters "John Doe" in the Full Name field.
3. User enters "john.doe@example.com" in the Email Address field.
4. User enters "password123" in the Password field.
5. User enters "password123" in the Confirm Password field.
6. User enters "1990-01-01" in the Date of Birth field.
7. User selects "Male" in the Gender field.
8. User selects "Yes" for the Newsletter Subscription.
9. User clicks the Submit button.

Expected Outcome:

- The form submits successfully.
- The user receives a success message or is redirected to a confirmation page.
- The entered data is saved to the server.

Positive Scenario 2: Successful Registration without Newsletter Subscription

Steps:

1. User navigates to the registration form page.
2. User enters "Jane Doe" in the Full Name field.
3. User enters "jane.doe@example.com" in the Email Address field.
4. User enters "P@ssword456" in the Password field.
5. User enters "P@ssword456" in the Confirm Password field.
6. User enters "1985-05-05" in the Date of Birth field.
7. User selects "Female" in the Gender field.
8. User leaves the Newsletter Subscription field unselected.
9. User clicks the Submit button.

Expected Outcome:

- The form submits successfully.
- The user receives a success message or is redirected to a confirmation page.
- The entered data is saved to the server.

Negative Scenarios

Negative Scenario 1: Submission with Invalid Email Address

Steps:

1. User navigates to the registration form page.
2. User enters "John Doe" in the Full Name field.
3. User enters "john.doe.com" in the Email Address field.
4. User enters "password123" in the Password field.
5. User enters "password123" in the Confirm Password field.
6. User enters "1990-01-01" in the Date of Birth field.
7. User selects "Male" in the Gender field.
8. User selects "Yes" for the Newsletter Subscription.

9. User clicks the Submit button.

Expected Outcome:

- The form does not submit.
- An error message is displayed indicating that the email address is invalid.
- The user is prompted to correct the email address.

Negative Scenario 2: Submission with Mismatched Password and Confirm Password

Steps:

1. User navigates to the registration form page.
2. User enters "Anna Maria" in the Full Name field.
3. User enters "anna@example.com" in the Email Address field.
4. User enters "password123" in the Password field.
5. User enters "password12" in the Confirm Password field.
6. User enters "1985-05-05" in the Date of Birth field.
7. User selects "Female" in the Gender field.
8. User selects "No" for the Newsletter Subscription.
9. User clicks the Submit button.

Expected Outcome:

- The form does not submit.
- An error message is displayed indicating that the password and confirm password fields do not match.
- The user is prompted to correct the confirm password field.

Test Writing:

a. Write automated test scripts (using a language and testing framework of your choice, e.g Cypress for JavaScript) for the positive scenarios identified in the previous step. Ensure that the scripts cover the key functionalities of the registration form.

Positive Scenario 1: Successful Registration with Valid Data

JS codeL

```
describe('Registration Form - Positive Scenario 1', () => {
  it('should successfully submit the form with valid data', () => {
    cy.visit('http://example.com/registration'); // Replace with the actual URL
    cy.get('input[name="fullName"]').type('John Doe');
    cy.get('input[name="email"]').type('john.doe@example.com');
    cy.get('input[name="password"]').type('password123');
    cy.get('input[name="confirmPassword"]').type('password123');
    cy.get('input[name="dateOfBirth"]').type('1990-01-01');
    cy.get('select[name="gender"]').select('Male');
    cy.get('input[name="newsletter"][value="Yes"]').check();
    cy.get('button[type="submit"]').click();
    // Assert the form submission was successful
    cy.url().should('include', '/confirmation'); // Replace with the actual confirmation page URL
    cy.contains('Registration successful').should('be.visible');
  });
});
```

Positive Scenario 2: Successful Registration without Newsletter Subscription

JS code:

```
describe('Registration Form - Positive Scenario 2', () => {
  it('should successfully submit the form without selecting newsletter subscription', () => {
    cy.visit('http://example.com/registration'); // Replace with the actual URL
    cy.get('input[name="fullName"]').type('Jane Doe');
    cy.get('input[name="email"]').type('jane.doe@example.com');
```

```
    cy.get('input[name="password"]').type('P@ssword456');
    cy.get('input[name="confirmPassword"]').type('P@ssword456');
    cy.get('input[name="dateOfBirth"]').type('1985-05-05');
    cy.get('select[name="gender"]').select('Female');
    // Do not select the newsletter subscription
    cy.get('button[type="submit"]').click();
    // Assert the form submission was successful
    cy.url().should('include', '/confirmation'); // Replace with the actual confirmation page URL
    cy.contains('Registration successful').should('be.visible');
  });
});
```

Notes:

- Replace http://example.com/registration with the actual URL of your registration form.
- Adjust the form field selectors (input[name="fullName"], input[name="email"], etc.) to match the actual HTML structure of your form.
- Adjust the URL and success message assertions as per your application's specific implementation.

How to Run These Tests:

1. **Install Cypress**: Ensure you have Node.js installed, then install Cypress via npm:

```
npm install cypress --save-dev
```

2. **Set Up Cypress**: Open Cypress for the first time, which will create the necessary folder structure:

```
npx cypress open
```

3. **Create Test Files**: Save the above test scripts in the cypress/integration directory, with appropriate file names (e.g., registration-positive-scenarios.spec.js).

4. **Run Tests**: Open Cypress and run the tests through the Cypress Test Runner:

```
npx cypress open
```