**TEST TASK**

**NAME : INSHA BARI**

**COMPANY : ELITE IT TEAM**

**POSITION : JUNIOR SQA ENGINEER**

Documentation and Test Scenario Identification

### Test Plan Outline for Registration Form

#### 1. Introduction

This document contains the test plan for the website's online registration form.The purpose of this test plan is to define the strategy and approach for testing the registration form to ensure it functions correctly and meets the specified requirements.

#### 2. Objectives

* To verify that the registration form collects and validates user information accurately.
* To guarantee that incorrect inputs are accompanied by error notifications.
* To verify the server's successful receipt of the data after the form has been correctly filled out.
* **3. Scope:**

The scope of this test is to check the user interface and ensuring that the registration form functions properly. I am double-checking the input fields for Full Name, Email Address, Password, Confirm Password, Date of Birth, Gender, and Newsletter Subscription to ensure that they are all valid. Additionally, I am verifying the form submission and awaiting the server response.

**4. Test Environment**

IDE: Pycharm

Devices: Desktop

OS: Windows

Tools: Pytest

**5. Test Data**

Full Name, Email Address, Password, Confirm Password

Various Date of Birth inputs

Different Gender selections

Newsletter Subscription choices

#### 6. Test Scenarios

* Positive scenarios: Successful form submission with valid data.
* Negative scenarios: Messages that indicate errors for inputs that are not valid.

#### 7.Test Execution Schedule

| Activity | Start Date | End Date |
| --- | --- | --- |
| Test Planning | 2024-07-15 | 2024-07-16 |
| Test Case Development | 2024-07-17 | 2024-07-18 |
| Test Environment Setup | 2024-07-19 | 2024-07-20 |
| Test Execution | 2024-07-21 | 2024-07-23 |
| Defect Reporting | 2024-07-21 | 2024-07-23 |
| Test Completion | 2024-07-24 | 2024-07-25 |

#### 8. Risks and Assumptions

* Assumptions:
* The user interface of the registration form is stable.
* The test setting is similar to the environment.

Risks:

* Changes in form design during the test cycle.
* Inconsistent performance on various devices and browsers

**Test Cases**

RF \_01  
**Test Case Description:** Validate successful form submission with all valid inputs.  
**Preconditions:** User is on the registration form page.  
**Test Steps:**

1. Enter a valid Full Name.

2nd Enter a valid Email Address.

3rd Enter a Password (at least 8 characters).

4th Confirm the Password.

5th Enter a valid Date of Birth.

6th Select a Gender.

7th Choose Yes for Newsletter Subscription.

8th Click the Submit Button.

**Expected Results:**

* The form is submitted successfully.
* User receives a confirmation message.
* Data is sent to the server.  
  **Post-conditions:** User is redirected to the confirmation page.

#### Test Case 2

RF\_02  
**Test Case Description:** Validate error message for mismatched passwords.  
**Preconditions:** User is on the registration form page.  
**Test Steps:**

1. Enter a valid Full Name.
2. Enter a valid Email Address.
3. Enter a Password (at least 8 characters).
4. Enter a different Confirm Password.
5. Enter a valid Date of Birth.
6. Select a Gender.
7. Choose Yes for Newsletter Subscription.
8. Click the Submit Button.  
   **Expected Results:**

* An error message is displayed indicating the passwords do not match.
* The form is not submitted.  
  **Post-conditions:** User remains on the registration form page.

#### Test Case 3

RF\_03  
**Test Case Description:** Validate error message for invalid email address.  
**Preconditions:** User is on the registration form page.  
**Test Steps:**

1. Enter a valid Full Name.
2. Enter an invalid Email Address.
3. Enter a Password (at least 8 characters).
4. Confirm the Password.
5. Enter a valid Date of Birth.
6. Select a Gender.
7. Choose Yes for Newsletter Subscription.
8. Click the Submit Button.  
   **Expected Results:**

* An error message is displayed indicating the email address is invalid.
* The form is not submitted.  
  **Post-conditions:** User remains on the registration form page.

## Scenario Identification

### Positive Scenarios

**Successful Registration**

**Steps:** Complete all fields with accurate data, then submit the form.

**Expected Outcome:** Form submits successfully, user receives a confirmation message, and data is sent to the server.

**Subscription to Newsletter**

**Steps:** successful registration snd subscription to news letter.

**Expected Outcome:** Form submits successfully, user receives a confirmation message including newsletter subscription confirmation, and data is sent to the server.

### Negative Scenarios

**Mismatched Passwords**

**Steps:** Complete all fields with valid data, excluding Password and Confirm Password (which do not match), then click "Submit" to send the form.

**Expected Outcome:** Error message indicating that passwords do not match and form is not submitted.

**Invalid Email Address**

**Steps:** Enter valid data for all fields except Email Address (use an incorrect email format) and submit the form.

**Expected Outcome:**The email address is invalid and that the form has not been submitted.

**Testing scripts**

**Language: Python**

**Tool used: Pytest**

**Description:** This automated test script creates a test application named ‘app’ and assumes it has a registration form (with functional backend) with the above mentioned fields.

**import pytest**

**from my\_app import create\_app, db**

**from my\_app.models import User**

**@pytest.fixture**

**def app():**

**app = create\_app(test\_config=True)**

**with app.app\_context():**

**db.create\_all()**

**yield app**

**db.drop\_all()**

**@pytest.fixture**

**def client(app):**

**return app.test\_client()**

**def test\_registration\_form\_valid\_without\_newsletter(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'N',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'Registration successful' in response.data**

**def test\_registration\_form\_valid\_with\_newsletter(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'Y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'Registration successful and Subscribed to Our Newletter' in response.data**

**def test\_registration\_form\_missing\_FullName(client):**

**response = client.post('/register', data={**

**'Full Name': '',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'This field is required' in response.data**

**def test\_registration\_form\_invalid\_email(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'invalid-email',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'Invalid email address' in response.data**

**def test\_registration\_form\_password\_mismatch(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'differentpassword',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'Field must be equal to password' in response.data**

**def test\_registration\_form\_short\_password(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'short',**

**'Confirm Password': 'short',**

**'Date of Birth': '2000-01-01',**

**'Gender': 'M',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'Field must be at least 8 characters long' in response.data**

**def test\_registration\_form\_missing\_date\_of\_birth(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '',**

**'Gender': 'M',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'This field is required' in response.data**

**def test\_registration\_form\_missing\_gender(client):**

**response = client.post('/register', data={**

**'Full Name': 'Test User',**

**'Email Address': 'testuser@example.com',**

**'Password': 'password123',**

**'Confirm Password': 'password123',**

**'Date of Birth': '2000-01-01',**

**'Gender': '',**

**'Newsletter Subscription': 'y',**

**'Submit': 'Submit'**

**}, follow\_redirects=True)**

**assert b'This field is required' in response.data**