

CEEZER Task Solution (Test Strategy)

1. Introduction

This document demonstrates the test approach, framework, and methodologies to write end-to-end (E2E) tests for the signup process using Cypress. The goal was to create a comprehensive test suite that validates functionality, error handling, accessibility, and visual correctness, covering both happy and edge cases.

2. Test Objectives

The main objectives of the test strategy are:

1. To validate that signup process functions as expected with valid data and invalid data.
2. To validate that the application properly handles and displays error messages when invalid data is entered.
3. To confirm the application's resilience by simulating server errors and network delays.
4. To assess the accessibility of the signup flow, ensuring compliance with accessibility standards.
5. To detect any UI regressions by using visual testing.

3. Test Scope

- E2E testing of the signup flow, focusing on field validation with valid & invalid test data, user navigation, error handling, and successful signup completion.
- Visual regression testing using Percy for all critical screens of the signup flow.
- Accessibility testing at each step of the signup flow using axe-core.
- Error scenario handling using Cypress intercept for simulating server errors and network conditions.

4. Test Approach & Testing Types

1. Functional Testing:

- Validation of the core signup functionality, including field validation, successful signup, and error message handling.

2. Negative Testing:

- Testing with invalid data (e.g., wrong email format, missing required fields) to ensure proper validation and feedback mechanisms.

3. Error Handling:

- Simulate network delays and server-side errors (500 Internal Server Error) to verify graceful degradation of the application and user notification.

4. Accessibility Testing:

- Utilize axe-core to identify critical accessibility violations, focusing on WCAG 2.0 standards, covering both serious and critical issues.

5. Visual Regression Testing:

- Capture visual snapshots at each form step and during error states to ensure that no unintended UI changes occur over time.

5. Test Scenarios Covered:

1. Valid Signup Flow
2. Invalid Input Handling
3. Error Handling with Server Delays
4. Simulate Server Error
5. Accessibility Testing
6. Visual Testing

6. Test Cases

Test Case 1: Valid Signup Flow

- **Objective:** To verify that the user can successfully complete the signup process using valid data.
- **Preconditions:** The signup page is accessible.
- **Steps:**
 1. Navigate to the signup page.
 2. Fill out the "User Details" form with valid information e.g name, email etc
 3. Click on Signup button to proceed to the "Company Details" form.
 4. Fill out the company information with the valid data e.g company name etc
 5. Click create account.
 6. Verify that the user is redirected to the confirmation/success page.
- **Expected Result:** The user should be successfully signed up and redirected to the success page.
- **Actual Result:** User are able to sign up successfully.

Test Case 2: Empty Required Fields

- **Objective:** To validate that appropriate error messages are displayed when required fields are left empty.

- **Preconditions:** The signup page is accessible.
- **Steps:**
 1. Navigate to the signup page.
 2. Leave the "First Name" and "Email" fields empty.
 3. Click "Next" to proceed.
 4. Verify that an error message is displayed under each empty required field (e.g., "please enter your first name etc).
 5. Fill in the "Full Name" field, but leave the "Email" field empty.
 6. Click "Next" again.
 7. Verify that only the error for the "Email" field is shown.
 8. Run an accessibility check with axe-core.
- **Expected Result:** Error messages should appear when required fields are empty, and the user should not be able to proceed.
- **Actual Result:** User are able to see the errors of mandatory fields.

Test Case 3: Invalid Email Format

- **Objective:** Verify that the signup process does not accept invalid email formats.
- **Preconditions:** The signup page is accessible.
- **Steps:**
 1. Navigate to the signup page.
 2. Fill in the "Company Name"
 3. In the "Email" field, enter the following invalid email formats one by one and try to proceed:
 - john.doe
 - john@doe
 - john.doe@
 - john.doe@example
 4. After each invalid email entry, attempt to click "Signup button."
 5. Verify that an appropriate error message is displayed for each invalid email format (e.g., "Please enter a company email address").
- **Expected Result:** User should not allow to submit with invalid email formats. Error messages should be displayed.

- **Actual Result:** User are able to see error message.

Test Case 4: Simulating Network Delays

- **Objective:** Verify that the application handles network delays gracefully.
- **Preconditions:** The signup page is accessible.
- **Steps:**
 1. Navigate to the signup page.
 2. Fill in all required fields with valid data.
 3. Use Cypress `cy.intercept()` to introduce a 5-second network delay when the user submits the form.
 4. Submit the form.
 5. Verify that the form submission is delayed, but no error messages are displayed.
 6. Once the delay is over, verify that the user is redirected to the success page.
- **Expected Result:** The application should display a loading indicator during the network delay and redirect the user to the success page once the delay ends.
- **Actual Result:** Users are able to redirect successfully.

Test Case 5: Verify the signup flow with Project developer carbon credit

- **Objective:** verify that the user can successfully complete the signup process using Project developer valid data.

Test Case 6: Simulating Server Error (500 Internal Server Error)

- **Objective:** To verify that the application handles server errors and displays a meaningful error message to the user.

Test Case 7: Accessibility Validation (axe-core)

- **Objective:** To validate that the signup process complies with accessibility standards at every step.

7. Tools & Framework

- **Cypress:** For writing and executing E2E tests with POM approach.
- **Mochawesome Reporter:** For generating detailed test reports.
- **Percy:** For visual regression testing.
- **Axe-core:** For accessibility testing, ensuring that the UI complies with accessibility standards.

Test Automation Framework

- **Page Object Model (POM):** Implements separation of concerns between test logic and UI interaction. All page elements and actions are encapsulated within reusable page object files.
- **Data-Driven Testing:** Using Cypress fixtures for managing both valid and invalid input data. This allows testing multiple scenarios with minimal code changes.

8. Test Reporting

- **Mochawesome** generates detailed reports that show the test results, including failed/passed assertions, and includes screenshots for visual validation.
- **Percy Dashboard** is used to review visual snapshots and identify any UI regressions.

9. Conclusion

The tests are designed to cover the key aspects of functionality, error handling, accessibility, and UI consistency. The use of POM ensures maintainability and scalability, while tools like Percy and axe-core ensure that visual and accessibility standards are maintained across the signup flow.

Bug Report:

Bug Report #1: Missing Validation on Empty Fields in Signup Form

- Bug ID: BR-001
- Reported By: Muhammad
- Date Reported: 08/09/24
- Severity: Medium
- Priority: Medium
- Status: Open
- **Environment:** Staging (<https://staging.ceezer.com/sign-up/create>)
- **Description:** When the user clicks on the **signup button** without filling in any details, an error message asking to enter details should appear. Once the user starts entering valid data in any field (e.g., name, email) and switch to new field, then error message should disappear, but this validation is missing.

Steps to Reproduce:

1. Navigate to the Signup page.
2. Leave all fields empty and click the Signup button.
3. Enter data into any field (e.g., Name or Email).
4. Observe the behavior.

- **Expected Result:**

- Error message should appear when clicking the signup button with empty fields. That's correct.
- Once valid details are entered into the form fields, the error message should disappear.

- **Actual Result:**

- User are still able to see error message on the input field. Field validation is missing.

- **Suggested Fix:**

- Implement field-level validation for each required field (name, email, company, etc.).
- Add an error message on each field when it is left blank and clicked on the submit button.
- Disable the **Signup** button until all required fields are filled with valid data.

Bug Report #2: Missing Validation of fields (numbers or special characters are allowed).

- **Bug ID:** BR-002
- **Reported By:** Muhammad
- **Date Reported:** 08/09/2024
- **Severity:** Medium
- **Priority:** Medium
- **Status:** Open
- **Environment:** Staging
- **Description:** There is no validation in fields (e.g **City** and **Street** fields), allowing users to input numbers or non-appropriate values (e.g., a numeric street name) into fields that are meant to accept alphabetic input only.
- **Steps to Reproduce:**
 1. Navigate to the **Signup Form**- Company details form.

2. Enter numeric or other invalid characters into the **City** and **Street** fields.
3. Proceed with the form submission.

- **Expected Result:**

- Fields like City and Street should only accept alphabetic input and provide validation for incorrect values (e.g., if numeric input is entered).

- **Actual Result:**

- Numeric input is accepted in the **City** and **Street** fields without any validation or error message.

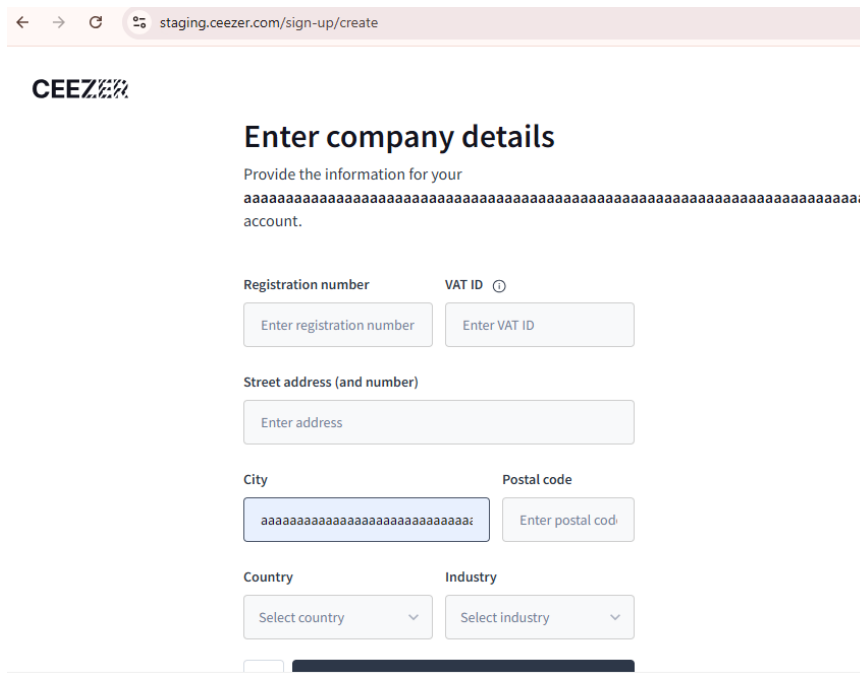
- **Suggested Fix:**

- Implement validation rules for City and Street fields to only accept alphabetic characters.
- Display an error message when numeric or invalid input is entered.

Bug Report #3: No Validation for Excessively Long Inputs

- Bug ID: BR-003
- Reported By: Muhammad
- Date Reported: 08/09/2024
- Severity: Medium
- Priority: Medium
- Status: Open
- Environment: Staging
- Description: Currently, form fields do not restrict the length of user input, allowing excessively long entries. This can lead to backend errors, UI breaks, or unintended behaviors.
- **Steps to Reproduce:**
 1. Navigate to the signup form.
 2. Enter an excessively long string in the Name, Street, or City fields.
 3. Submit the form.
- **Expected Result:**
 - A maximum length for each field should be enforced (e.g., 50 characters for names and 100 characters for addresses).
 - Error messages should appear if a user exceeds the character limit.

- Actual Result:
 - There are no restrictions, and excessively long inputs are accepted.
- Suggested Fix:
 - Add a maximum character length for each form field.
 - Validate the form and display error messages for long inputs.
- Attachments:



The screenshot shows a web browser window with the address bar displaying "staging.ceezer.com/sign-up/create". The page features the CEEZER logo and a heading "Enter company details". Below the heading, a subtext reads "Provide the information for your account." followed by a line of 30 'a' characters. The form contains several input fields: "Registration number" and "VAT ID" (with a help icon), "Street address (and number)", "City" (containing 30 'a' characters), "Postal code", "Country" (a dropdown menu), and "Industry" (a dropdown menu). A progress bar at the bottom indicates the current step in the registration process.

Bug Report #4: No Validation for Special Characters in Form Fields

- Bug ID: BR-004
- Reported By: Muhammad
- Date Reported: 08/09/2024
- Severity: Medium
- Priority: Medium
- Status: Open
- Environment: Staging
- Description: Special characters (e.g., @, #, \$, etc.) are allowed in all fields such as Name, Street, and City without any validation, which is not typical for these types of inputs.

- **Steps to Reproduce:**

1. Navigate to the signup form.
2. Enter special characters (e.g., @John, #City123) in fields like Name, Street, and City.
3. Proceed with the form submission.

- **Expected Result:**

- Form fields like Name and City should not allow special characters and should display a validation error when they are inputted.

- **Actual Result:**

- Special characters are accepted in form fields without validation errors.

- **Suggested Fix:**

- Implement a regex validation for special characters in fields where only alphabetic input is allowed (e.g., Name, City).
- Provide error messages if a user attempts to input special characters.

Thanks!