# Test Approach Document for Ensek Test Website

Table of Contents

[Test Approach Document for Ensek Test Website 1](#_Toc178456970)

[Introduction 2](#_Toc178456971)

[Objective 2](#_Toc178456972)

[Scope of Testing 2](#_Toc178456973)

[Test Strategy 2](#_Toc178456974)

[Manual Testing Scope 2](#_Toc178456975)

[1.Unit Testing: 2](#_Toc178456976)

[2.Functional Testing: 2](#_Toc178456977)

[3.Performance Testing 3](#_Toc178456978)

[4.User Experience Testing: 3](#_Toc178456979)

[5.Cross-browser Testing: 3](#_Toc178456980)

[6.Exploratory Testing: 3](#_Toc178456981)

[Automation Testing Scope 3](#_Toc178456982)

[UI Automation (Selenium with C#): 3](#_Toc178456983)

[1.Automate main user validation functionality: 3](#_Toc178456984)

[2.Automate validation for error scenarios 3](#_Toc178456985)

[3.Regression Testing: 4](#_Toc178456986)

[4.Data-driven Testing: 4](#_Toc178456987)

[5.Cross-browser Testing: 4](#_Toc178456988)

[6.Performance Testing: 4](#_Toc178456989)

[Test Tools and Frameworks 4](#_Toc178456990)

[Test Environments 4](#_Toc178456991)

[Test Execution Schedule 5](#_Toc178456992)

[Entry and Exit Criteria 5](#_Toc178456993)

[Entry Criteria: 5](#_Toc178456994)

[Exit Criteria: 5](#_Toc178456995)

[Risk and Mitigation 5](#_Toc178456996)

[Reporting and Metrics 5](#_Toc178456997)

# Introduction

This document outlines the test approach for the Ensek Automation Candidate Test Website focusing on both manual and automation testing. As the Lead QA Engineer, I will ensure that all functionality is tested comprehensively to ensure quality, performance, and usability across the system.

## Objective

The primary objective of this test approach is to:

* Verify that the system meets the unit testing, functional testing, performance testing and user experience testing requirements.
* Ensure the user interface is intuitive and error-free.
* Automate repetitive tests to enhance testing efficiency.

## Scope of Testing

This document covers both manual and automation testing for the following areas:

* Unit Testing
* Functional Testing
* Performance Testing
* User Experience Testing
* Cross-browser Testing

# Test Strategy

## Manual Testing Scope

Manual testing will focus on areas such as UI/UX, error handling, and exploratory testing. The following areas are covered:

### 1.Unit Testing:

* Launching the application
* Verifying all menu options
* Verifying all page navigation
* Verifying the Button that performs the necessary action
* Verifying the links that navigates the desired page

### 2.Functional Testing:

* Every Page Verification
* Registration Functionality
* Login Functionality
* Validate buy energy
* Error messages for invalid inputs.

### 3.Performance Testing

* Verify the application performance with respect to the Load and stress

### 4.User Experience Testing:

* Verify layout and design across various devices.
* Ensure consistency in design elements, such as buttons, forms, and navigation.
* Test accessibility standards including the typo

### 5.Cross-browser Testing:

* Ensure compatibility across different browsers (Chrome, Firefox, Safari, Internet Explorer, Edge).
* Test for responsiveness on different screen sizes.

### 6.Exploratory Testing:

* Perform exploratory testing to find edge cases and usability issues that may not be covered in predefined test cases.

*NOTE: Kindly refer the Excel Document “Ensek – Test Plan Document for Manual testing” which has 3 sheets named Test Coverage, Test Cases and Bug Lists.*

* *Test Coverage Sheet – Overview of number of test cases identified which includes the details of priority, pass and failure test cases.*
* *Test Case Sheet – Test cases identified with respect to Unit Testing, Functional Testing, Performance Testing, User Experience Testing, Cross Browser Testing and Exploratory Testing*
* *Bug List – Bugs Identified while performing the manual testing which includes the screenshots for the issues.*

## Automation Testing Scope

Automated testing will cover repetitive tasks and key functional flows. The following areas are covered:

## UI Automation (Selenium with C#):

### 1.Automate main user validation functionality:

Launch, registration, login, navigating to menu options Home, About, Contact, navigating to pages Buy Energy, Sell Energy and Learn More and validating the Buy energies.

### 2.Automate validation for error scenarios

Invalid credentials error while registering and while login attempt

### 3.Regression Testing:

Develop automated regression suites that can be triggered with every release to ensure existing functionality is not broken.

*(Note: I did not add these into the automation test package)*

### 4.Data-driven Testing:

Automate buying energy scenarios, and form inputs using a data-driven approach to cover multiple sets of input combinations.

*(Note: I have covered only high-level cases)*

### 5.Cross-browser Testing:

NUnit Tests to perform cross browser testing to execute test cases across multiple browsers automatically.

*(Note: For time Being I have used only Chrome Browser)*

### 6.Performance Testing:

Performance testing will ensure the website handles load and responds within acceptable limits:

Load Testing: Simulate multiple users accessing the site concurrently using tools like JMeter.

Stress Testing: Identify the maximum capacity of the website by incrementally increasing the load until the system breaks.

Response Time Testing: Validate page load times for critical actions such as logging in, Buy energy.

(*Note*: I did not include the performance testing script in the automation package)

NOTE: Kindly refer the Git Repository for the automation test package which includes all the above listed automation testing in it.

Git Repository Link:

## Test Tools and Frameworks

* Manual Testing: TestRail (for test case management), Jira (for defect tracking).
* Automation Framework: Selenium with C# for UI automation, NUnit for unit and integration testing.
* Performance Testing: Apache JMeter.
* Cross-browser Testing: NUnit.

## Test Environments

Testing will be conducted across multiple environments:

Staging Environment: To replicate production-like scenarios before release.

Local Development Environment: For continuous integration testing during development.

## Test Execution Schedule

Phase 1: Manual functional and UI/UX testing (1weeks).

Phase 2: Automation script development and execution (1-3 weeks).

Phase 3: Performance and UAT testing (1 week).

Phase 4: Regression and cross-browser testing (ongoing for every release).

## Entry and Exit Criteria

### Entry Criteria:

* Application is deployed in a stable environment.
* All required test data is available.

### Exit Criteria:

* All high-priority test cases have been executed and passed.
* All critical defects are resolved or deferred with approval.
* The automation regression suite has passed successfully.

## Risk and Mitigation

Risk: Browser compatibility issues on less popular browsers.

Mitigation: Prioritize testing on major browsers, and only run smoke tests on less popular ones.

## Reporting and Metrics

Daily and weekly test execution reports will be provided, including:

* Test case pass/fail rates which will be updated in the test plan document sheet “Test Coverage”
* Defect density and severity distribution which will be updated in the Test Plan sheet “Bug List”
* Performance benchmarks.