Test Design Specification

Table of Contents

1 Introduction 2

2 Test Objective 2

3 Detailed Testing Strategy 2

3.1 Unit Testing 2

3.1.1 White Box Testing 2

3.1.2 Black Box Testing 3

3.2 Integration Testing 3

3.2.1 Incremental Testing 3

3.3 System Testing 3

3.3.1 Function Validation Testing 4

3.3.2 Performance testing 4

3.3.3 Compatibility Testing 4

3.3.4 Accessibility Testing 5

4 Pass/Fail Criteria 5

1 Introduction

This document provides the test documentation that will facilitate the technical tasks of testing including the detailed test cases for both white box and black box testing. Each test case specifies who will be performing the test, the preconditions required to execute each test case, the specific item to be tested, the input, expected output or results, and procedural steps where applicable.

2 Test Objective

The objective of this document is to expand on the test plan and provide specific information needed to perform the necessary tests. By providing detailed test information, we hope to reduce the probability of overlooking items and improve test coverage. Testers will be able to use each test case provided in this document to move forward and begin testing. Test results will be logged in an excel.

3 Detailed Testing Strategy

3.1 Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test functions or code modules. The unit test cases shall be designed to test the validity of the program's correctness.

3.1.1 White Box Testing

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. The test cases that have been generated shall cause each condition to be executed at least once. To ensure this happens, we are applying Basis Path Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply.

Each function of the backend is executed independently; therefore, a program flow for each function has been derived from the code. The development team will be performing all white box testing.

3.1.2 Black Box Testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. Black box testing will be performed by the test team.

All procedural steps have been included to assist the team in executing the various tests.

3.2 Integration Testing

3.2.1 Incremental Testing

There are two primary modules that will need to be integrated: the UI Interface and the backend API. The two components, once integrated, will form the complete Application testing. The following describes these modules as well as the steps that will need to be taken to achieve complete integration. We will be employing an incremental testing strategy to complete the integration. The integration testing will be performed by the testing team.

Module 1 - UI

This module provides a simple GUI where the user can perform the different actions (functions). This module will be tested separate from the backend to check if each interface (e.g., create button) is functioning properly, and in general, to test if the mouse-event actions are working properly. The testing will be performed by writing a selenium test for each component.

Module 2 – API

The API part will be tested separately and thoroughly to verify the functionality and to verify it under different test data. For example, Create Book need to be tested with providing mandatory values and with not providing mandatory values. Also, need to verify different data types as well. API will be tested separately using RestAssured and Postman + Newman.

3.3 System Testing

The goals of system testing are to detect faults that can only be exposed by testing the entire integrated system or some major part of it. Generally, system testing is mainly concerned with areas such as performance, security, validation, load/stress, and

configuration sensitivity. But in our case well focus only on function validation and performance. And in both cases, we will use the black-box method of testing.

3.3.1 Function Validation Testing

The integrated Application will be tested based on the requirements to ensure that we built the right application. In doing this test, we will try to find the errors in the inputs and outputs, that is, we will test each function to ensure that it properly implements the requirements.

Function	**Expected Behavior**
Load	Should load existing books and authors
Create	Should create and store new books and authors
Edit	Should update existing books and authors
Delete	Should remove existing books and authors upon deleting
Search	Should search and show the results accordingly

3.3.2 Performance testing

This test will be conducted to evaluate the fulfillment of a system with specified performance requirements. It will be done using black box testing method. And this will be performed by:

- Storing the maximum data in the DB and trying to insert and observe how the application will perform when it is out of boundary.
- Deleting data and check if it follows the right sorting algorithm to sort the resulting data or output.
- Trying to store new data and check if it overwrites the existing once.
- Trying to load the data while they are already loaded

3.3.3 Compatibility Testing

This testing will be covered to see how the application will behave in different environments. Customers will use different Oss and browsers. So, from the testing point of view we also need to cover some part of these combinations. For this test we will use following combinations.

- Windows – Chrome, Firefox, Edge

3.3.4 Accessibility Testing

Accessibility testing should be done since this application is having a UI. We need to make sure that any differently abled customer also can use the 100% features of the system without any issue. Following third party tools will be used to do the accessibility testing.

- WAVE
- AXE
- NVDA
- CCA

4 Pass/Fail Criteria

This section will include the master list of both white box and black box tests which will be used to track the progress of the testing. A test will be considered a failure if the expected result or output is not achieved. A bug report will be filled out for each failure and will be submitted to the development team for correction. After the bug has been fixed, the test case will be repeated.