**Test Plan**

Stuart Allender, Sonia Ali, William Ashworth, Minal Patel, Tom Smith, Jevale Sterling, Jeremy Sullivan and Alex Thain

Contents

[1. Introduction 3](#_Toc2253920)

[1.1 Objectives 3](#_Toc2253921)

[1.2 Purpose 3](#_Toc2253922)

[1.3 Scope 3](#_Toc2253923)

[1.3.1 In Scope 3](#_Toc2253924)

[1.3.2 Out of Scope 3](#_Toc2253925)

[1.4 Project Overview 3](#_Toc2253926)

[2. Test Strategy 4](#_Toc2253927)

[2.1 Test Conditions 4](#_Toc2253928)

[2.2 Test Scenarios 4](#_Toc2253929)

[2.3 Test Constraints 4](#_Toc2253930)

[3. Test Approach/Methodology 4](#_Toc2253931)

[3.1 Overview 4](#_Toc2253932)

[3.1.1 Usability Testing 4](#_Toc2253933)

[3.1.2 Unit testing 5](#_Toc2253934)

[3.1.3 Regression Testing 5](#_Toc2253935)

[3.1.4 Performance Testing 5](#_Toc2253936)

[3.1.5 Functional Testing 5](#_Toc2253937)

[4. Test Management Process 6](#_Toc2253938)

[4.1 Test Management Tools 6](#_Toc2253939)

[4.2 Test Risks and Mitigation Factors 6](#_Toc2253940)

[5. Execution Strategy 7](#_Toc2253941)

[5.1 Entry and Exit Criteria 7](#_Toc2253942)

[5.2 Testing Life Cycle 7](#_Toc2253943)

[7](#_Toc2253944)

[6. Test Environment 7](#_Toc2253945)

[6.1 Testing Environment 7](#_Toc2253946)

[7. Milestones/Deliverables 8](#_Toc2253947)

[7.1 Test Schedule 8](#_Toc2253948)

[7.2 Deliverables 9](#_Toc2253949)

# 1. Introduction

This Test Plan has been created to communicate the test approach to all team members. The plan includes Objectives, Purpose, Assumptions and Risks, Testing Methodology, Test Management Process and the Test Environment. This test plan clearly documents the test deliverables and identifies the parameters both within and out of the scope.

## 1.1 Objectives

1. **Assure that the application meets the full requirements that have been set out including non – functional requirements and satisfies use case and user story scenarios.** *(At the end of the sprint cycle, the user should find that the project has met the expectations in relation to the requirements provided. Any changes will be documented and tested within the remaining time and within the ability of the team.)*
2. **Identify and expose all issues and associated risks, communicate all known issues to the team are addressed before realise.**  *(Ensure that all issues are addressed in a methodical testing manner of the application to ensure all issues(bugs) found are dealt with).*

## 1.2 Purpose

The Purpose of the Test Plan is to achieve the following:

1. Define testing strategies for each area to include all the functional and quality (non-functional) requirements.
2. Define bug-tracking procedures.
3. Identify testing risks.
4. Identify required resources and related information.
5. Provide testing Schedule.

## 1.3 Scope

### 1.3.1 In Scope

The test scope includes the following:

* Testing all functional, application performance, security and use case requirements listed in the use case scenarios
* End to End testing and testing interfaces of all systems that interact with the application

### 1.3.2 Out of Scope

The following is considered out of scope:

* Functional requirements testing for systems outside the application
* Testing of business SOP’s, disaster recovery and Business Continuing Plan.

## 1.4 Project Overview

The project is to produce a application to help with identifying threats. A word finding algorithm that will utilise words that are often used and flag them according to severity.

The application will have the following functions:

* Ability to upload singular or multiple files
* Ability to handle multiple data types e.g. PDF, Word, Excel
* Ability to Produce a business report to show overall
* Ability to upload Word lists
* Ability to Log in

# 2. Test Strategy

## 2.1 Test Conditions

**Example test condition:**  
In the above example, if we were to test the scenario 1, we can test the following conditions:  
1. Enter the country name as “India”(valid )and check for the addition of the country  
2. Enter a blank and check if the country gets added.  
In each case, the specific data is described and the goal of the test is much more precise.

## 2.2 Test Scenarios

1. Validate and advisor can log in
2. Validate an admin can log in
3. Validate if an existing world list can be updated
4. Validate an advisor can be downloaded
5. Validate an advisor can be shared
6. Validate an advisor/admin can upload file
7. Validate a admin can see a business report

## 2.3 Test Constraints

Constraints of Software testing:

1. No fixed approach in order to test the application. Every individual and plan their own test plan.
2. Time consuming as it takes 30–40% of SDLC life cycle.
3. You need to continuously learn and update your knowledge on multiple testing tools.

# 3. Test Approach/Methodology

## 3.1 Overview

The project is using an agile approach, with weekly iterations. At the end of each week the requirements identified for that iteration will be delivered to the team and will be tested.

This section will identify the what the testing methodologies are, how they will be used.

### 3.1.1 Usability Testing

Usability testing is to see how easy the application is to use by testing it with real users. It evaluates the ease of use. Users will be asked to complete tasks, while they are being observed. This will be to see where they encounter problems and experience confusion. If a problem or confusion is occurring consistently recommendations will be made on how to overcome these usability issues. TOOL

### 3.1.2 Unit testing

Unit testing is where individual unit/components of a software are tested. The purpose of this is to ensure/validate that each unit of the software performs as designed. Mocha is a Java-Script test framework. Mocha tests run serially, allowing for flexible and accurate reporting. Mocha can be used on various libraries such as should.js, express.js and chai. Mocha is an NPM package that allows you to write and execute unit tests for your JavaScript code. It uses assert statements, similar to JUnit in Java. Mocha gives you the ability to use “Describe” to organise your tests into headings and subheadings, and “It” to write individual tests in these areas.

Mocha runs in the command line, and lists your tests by Describe sections. It shows which individual tests passed, and which didn’t. You can set your own expected messages making it easy to see exactly which tests have failed.

You can also add the library “Chai”, allowing you to use a wider variety of assert statements, allowing you to perform different types of tests.

### 3.1.3 Regression Testing

Regression testing is the process of re-running tests when changes have been made to the application to make sure that pervious programming still works with new changes. The regression testing will take place when other tests have been run and code has been changed to fix bugs. Therefore, old tests will be re-run to ensure the application is still running with new code.

### 3.1.4 Performance Testing

Performance testing is to ensure software applications will perform well under their expected workload. This will look into the response time, reliability, resource usage and scalability. The goal is to eliminate performance bottlenecks rather than find bugs.

The Apache JMeter™ application is open source software Java application designed to load test functional behaviour and measure performance. Apache JMeter may be used to test performance both on static, dynamic resources and Web dynamic applications. It can be used to simulate a heavy load on a server, group of servers, network or object to test its strength or to analyse overall performance under different load types. We will use this to stress test our application, as part of our performance testing.

### 3.1.5 Functional Testing

Functional testing is when the application is tested against the functional requirements/ specifications. Functions are tested by feeding then inputs and inspecting the outputs. Functional testing ensures that the requirements are properly satisfied by the application.

Selenium is an open source tool which is used for automating the tests carried out on web browsers. Selenium runs the tests automatically to reduce the time required during regression testing. It also provides a playback tool for authoring functional tests without the need to learn a test scripting language (Selenium IDE).

It will be used to automate the testing/ navigation process through our application. With selenium we will make our tests further in depth with the use of Cucumber which is a feature file (given, when, then) based process that provides steps to test against when using our site as a User.

# 4. Test Management Process

## 4.1 Test Management Tools

A test management tool is a software used to manage test that have been previously specified by a test procedure.

QARA (Quality Assurance with Rapid Automation), a product by The Digital Group, is an integrated platform that supports both Manual and Automated Test Case Management. A declarative, no-scripting, no-coding functional test automation framework, its functionalities include test management, planning, and execution along with built-in reporting capabilities. This test management tool helps us maintain traceability throughout our project which means that all members of the team can see the work that’s been planned or already completed. We have seamlessly integrated QARA with JIRA so that all members of the team can keep tabs on the progress made.

## 4.2 Test Risks and Mitigation Factors

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Probability | Impact | Mitigation Plan |
| Test Manager has poor management skills | Low | Medium | Plan leadership for manager |
| Team members have a lack of required skills and knowledge for website testing | Medium | Medium | Plan training course to skill up your members |
| The project schedule is too tight – thus hard to complete. | High | High | Set test priority for each pf the test activity.  Testing team can control the preparation tasks (in advance) and the early communication with involved parties. |
| A lack of cooperation negatively affects the teams’ productivity | Medium | High | Encourage each team member in their tasks and inspire them to greater efforts |
| Wrong budget estimate and cost overruns | Medium | High | Establish this scope before beginning work, pay a lot of attention to project planning and constantly track and measure the progress |
| Defects found at a late stage of the cycle or at a late cycle are most likely to be due to unclear specifications and are time consuming to resolve. | Medium | High | Defect management plan is in place to ensure prompt communication and fixing of issues. |
| Scope completely defined | Medium | Medium | Scope is well defined but changes are in the functionality are yet not finalised or keep on changing |

# 5. Execution Strategy

## 5.1 Entry and Exit Criteria

Entry Criteria:

* Complete or partially testable code is available
* Requirements are defined and approved
* Availability of test data
* Test cases are developed and ready to use
* Test environment has been set up and all other necessary resources such as tool and devices are available

Exit Criteria:

* Execution of all test cases/conditions
* Desired and sufficient coverage of the requirements and functionalities under the test
* All the identified defects are corrected and closed
* No high priority or severity or critical bug has been left out
* Re – testing and closing all the high priority defects to execute corresponding regression scenarios successfully

## 5.2 Testing Life Cycle

Test Environment Set –up

Test Case Development

Requirement Analysis

Test Planning

Test Cycle Closure

Test Execution

# 6. Test Environment

## 6.1 Testing Environment

* Mocha – JS Test
* Selenium – Automation
* Jest – Integration (Coverage)
* Cypress.10 – End to End Complete Tool
* Cucumber – Automation
* J- Meter – Speed
* Postman – Endpoint Check
* QARA (Quality Assurance with Rapid Automation)

# 7. Milestones/Deliverables

## 7.1 Test Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task Name | Start | Finish | Time | Comments |
| Test Planning |  |  |  |  |
| Review Requirement Documents |  |  |  |  |
| First deploy to QA test environment |  |  |  |  |
| Unit Testing – Iteration 1 |  |  |  |  |
| Functional testing – Iteration 1 |  |  |  |  |
| Iteration 2 Deploy to QA test environment |  |  |  |  |
| Unit Testing – Iteration 2 |  |  |  |  |
| Functional testing – Iteration 2 |  |  |  |  |
| Regression Testing |  |  |  |  |
| UAT/Usability Testing |  |  |  |  |
| Resolution to final defects and final build testing |  |  |  |  |
| Deploy to staging environment |  |  |  |  |
| Performance Testing |  |  |  |  |
| Release |  |  |  |  |

## 7.2 Deliverables

|  |  |  |
| --- | --- | --- |
| Deliverable | For | Date/Milestone |
| Test Plan |  |  |
| Traceability Metrix |  |  |
| Test Results |  |  |
| Test Status Report |  |  |
| Metrics |  |  |

## 