Test Plan: 2CWK50

Version: 0.1.0

Date: March 2020

Authors: Ashley Williams, Smith Shodunke

# 1. Document Outline

This document describes the testing strategy for the 2CWK50 project

## 1.1 Contact

Please direct all questions and issues to the authors of this document via email.

# 2. Scope of the testing

## 2.1 In scope

* All of the functions within the simpleMaths.js library

## 2.2 Out of scope

* Any non-functional requirements such as stress and performance
* Any hardware involved

# 3. Testing type

All efforts will be focused on unit testing the individual components. Use and integration of the library with other components is out of the scope of this document.

# 4. Risks and Issues

“Risk is future’s uncertain event with a probability of occurrence and a potential for loss. When the risk actually happens, it becomes the ‘issue’.”

|  |  |
| --- | --- |
| Risk | Mitigation |
| Tester lacks the required skills for unit testing the library | Deliver testing workshop using Jest |
| Project schedule is too restricted | Prioritise testing activities |

# 5. Tests logistics

**Tests to be conducted by:** Smith Shodunke

**When will testing occur:** today (wc 16/3/2020)

## 5.1 Testing Objectives

Functions to test:

1. Login API
2. Get Chits from API
3. Search user APIII
4. Multiply

Goal: to write automated unit tests for each of the above functions.

## 5.2 Suspension Criteria

Automated tests will be written for all functions. Any failing tests will be relayed back to the development team for amendment.

## 5.3 Exit Criteria

Test phase will be deemed successful when 100% of test cases pass with 100% code coverage.

(Normally defined in terms of run rate and pass rate)

* Run rate: number of test cases executed/total test cases (Should be 100% unless a clear reason is provided)
* Pass rate: number of test cases passed/total cases executed (Depends on scope of project, but should be high)

# 5.4 Resource planning

(just some information, not relevant for this specific test plan)

HR resource:

1. Test manager – manage project/define direction
2. Tester – describes appropriate tools, techniques, technology, executes the tests and logs results
3. Developer in Test – writes the tests
4. Test administrator – maintains test environments (servers, DBs etc.)
5. SQA members – audits to ensure quality

System resource:

1. Server
2. Test tool
3. Network
4. Computer

## 5.5 Plan test environment

(just some information, not relevant for this specific test plan)

1. What architecture is required?
2. Does the company already have a test environment setup?
3. Is it populated with data?
4. What constraints does the environment have?

## 5.6 Schedule and Estimation

|  |  |  |
| --- | --- | --- |
| **Task** | **Members** | **Estimated effort** |
| **Create test specification** | Test designer (you) | 10 minutes |
| **Write the tests** | Test developer (also you) | 30 minutes |
| **Perform test execution** | Tester (also also you) | 2 minutes |
| **Test report** | Tester | 2 minutes |
| **Total** |  | **44 minutes** |

# 6. Test specification

**Exercise: Write the test specification for the simpleMaths.js library. For each of the four functions, we want to test for good data, erroneous data and boundary data.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Author** | **Function** | **Parameters** | **Expected** |
| 1 | Ash | Add() | Num1: 2  Num2: 2 | 4 |
| 2 | Ash | Add() | Num1: 0  Num2: 0 | 0 |
| 3 | Ash | Add() | Num1: -1  Num2: -1 | -2 |
| 4 | Ash | Add() | Num1: 9999999  Num2: 9999999 | 19999998 |

## 6.1 Assumptions

The test specification makes the following assumptions:

1. Negative numbers are allowed
2. Decimal numbers are allowed
3. The answer can be as large as 19,999,998
4. Anything that is not a number should throw an exception

# 7. Test deliverables

Before testing:

1. Test plan document
2. Test specifications table

During testing:

1. Test scripts
2. Test data
3. Error logs and execution logs

After testing:

1. Test report with run rate, pass rate and details of each test case
2. Test coverage report
3. Defect report
4. Test procedures guidelines
5. Release notes

# 6. Credit

The structure and content of this test plan has been influenced by <https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html>