

Project Title: System Verification and
Validation Plan for REVITALIZE

Author Name

October 31, 2022

1 Revision History

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

Contents

1	Revision History	i
2	Symbols, Abbreviations and Acronyms	iv
3	General Information	1
3.1	Summary	1
3.2	Objectives	1
3.3	Relevant Documentation	1
4	Plan	1
4.1	Verification and Validation Team	1
4.2	SRS Verification Plan	2
4.3	Design Verification Plan	2
4.4	Implementation Verification Plan	2
4.5	Automated Testing and Verification Tools	2
4.6	Software Validation Plan	2
5	System Test Description	3
5.1	Tests for Functional Requirements	3
5.1.1	Login Page Testing	3
5.1.2	Area of Testing2	6
5.2	Tests for Nonfunctional Requirements	6
5.2.1	Area of Testing1	7
5.2.2	Area of Testing2	7
5.3	Traceability Between Test Cases and Requirements	7
6	Unit Test Description	8
6.1	Unit Testing Scope	8
6.2	Tests for Functional Requirements	8
6.2.1	Module 1	8
6.2.2	Module 2	9
6.3	Tests for Nonfunctional Requirements	9
6.3.1	Module ?	9
6.3.2	Module ?	10
6.4	Traceability Between Test Cases and Modules	10

7	Appendix	11
7.1	Symbolic Parameters	11
7.2	Usability Survey Questions?	11

List of Tables

[Remove this section if it isn't needed —SS]

List of Figures

[Remove this section if it isn't needed —SS]

2 Symbols, Abbreviations and Acronyms

symbol	description
T	Test

[symbols, abbreviations or acronyms – you can simply reference the SRS (Author, 2019) tables, if appropriate —SS]

This document ... [provide an introductory blurb and roadmap of the Verification and Validation plan —SS]

3 General Information

3.1 Summary

[Say what software is being tested. Give its name and a brief overview of its general functions. —SS]

3.2 Objectives

[State what is intended to be accomplished. The objective will be around the qualities that are most important for your project. You might have something like: “build confidence in the software correctness,” “demonstrate adequate usability.” etc. You won’t list all of the qualities, just those that are most important. —SS]

3.3 Relevant Documentation

[Reference relevant documentation. This will definitely include your SRS and your other project documents (MG, MIS, etc). You can include these even before they are written, since by the time the project is done, they will be written. —SS]

Author (2019)

4 Plan

[Introduce this section. You can provide a roadmap of the sections to come. —SS]

4.1 Verification and Validation Team

[You, your classmates and the course instructor. Maybe your supervisor. You should do more than list names. You should say what each person’s role is for the project. A table is a good way to summarize this information. —SS]

4.2 SRS Verification Plan

[List any approaches you intend to use for SRS verification. This may just be ad hoc feedback from reviewers, like your classmates, or you may have something more rigorous/systematic in mind.. —SS]

[Remember you have an SRS checklist —SS]

4.3 Design Verification Plan

[Plans for design verification —SS]

[The review will include reviews by your classmates —SS]

[Remember you have MG and MIS checklists —SS]

4.4 Implementation Verification Plan

[You should at least point to the tests listed in this document and the unit testing plan. —SS]

[In this section you would also give any details of any plans for static verification of the implementation. Potential techniques include code walk-throughs, code inspection, static analyzers, etc. —SS]

4.5 Automated Testing and Verification Tools

[What tools are you using for automated testing. Likely a unit testing framework and maybe a profiling tool, like ValGrind. Other possible tools include a static analyzer, make, continuous integration tools, test coverage tools, etc. Explain your plans for summarizing code coverage metrics. Linters are another important class of tools. For the programming language you select, you should look at the available linters. There may also be tools that verify that coding standards have been respected, like flake9 for Python. —SS]

[The details of this section will likely evolve as you get closer to the implementation. —SS]

4.6 Software Validation Plan

[If there is any external data that can be used for validation, you should point to it here. If there are no plans for validation, you should state that here. —SS]

5 System Test Description

5.1 Tests for Functional Requirements

Subsections of the requirements will be divided into the events from our SRS, which are Login Page, Sign up Page, Main Page, Diet Section, Workout Section and Rest Section. There will be 1 test per functional requirement, and will follow the same order as functional requirements in SRS (ex. FR1 in VnV plan is the test for FR1 in SRS).

5.1.1 Login Page Testing

Testing all functional requirements for login page of REVITALIZE. (Refer to BE1 in SRS)

1. FR1

Control: Manual

Initial State: Loading stage of the login page

Input: An event that loads the login page

Output: Login page is displayed with all necessary components

Test Case Derivation: Request is made to load login page

How test will be performed: Tester will open REVITALIZE application and login page should be displayed

2. FR2

Control: Manual

Initial State: Login page is displayed with username textbox

Input: Enter username information in textbox

Output: Username information entered is displayed in textbox

Test Case Derivation: User can enter information in username textbox

How test will be performed: Tester will enter information in username textbox and checks if textbox displays what the tester entered.

3. FR3

Control: Manual

Initial State: Login page is displayed with password textbox

Input: Enter password information in textbox

Output: password information entered is displayed in textbox via hidden text

Test Case Derivation: User can enter information in password textbox

How test will be performed: Tester will enter information in password textbox and checks if textbox displays what the tester entered via hidden text.

4. FR4

Control: Manual

Initial State: Login page is displayed with login button

Input: Click login button

Output: Intended events occurs. Refer to FR9

Test Case Derivation: User clicks login button and a request is made based on username and password text-boxes

How test will be performed: Tester will click on login button and check if request is made correctly

5. FR5

Control: Manual

Initial State: Login page is displayed with forgot password button

Input: Click forgot password button

Output: Display forgot password screen with textbox to enter email

Test Case Derivation: User clicks forgot password button and request is made to display forgot password screen with textbox to enter email

How test will be performed: Tester will click on forgot password button and checks if forgot password screen is displayed with textbox to enter email

6. FR6

Control: Manual

Initial State: Login page is displayed with stay logged in checkbox that is empty

Input: Click stay logged in checkbox

Output: Display a check-mark in the stay logged in checkbox if checkbox is empty, else if checkbox contains check-mark already it will then display an empty checkbox

Test Case Derivation: User clicks on stay logged in checkbox and displays appropriate action

How test will be performed: Tester will click on checkbox and checks to see if check-mark is displayed if checkbox was empty and if an empty checkbox appears if checkbox contained a check-mark

7. FR7

Control: Manual

Initial State: Loading stage of REVITALIZE where previous state had stay logged in checkbox checked

Input: An event that loads REVITALIZE

Output: Display main page, with same data from previous state of main page

Test Case Derivation: User can load REVITALIZE and main page is displayed with same data as the previous time user opened REVITALIZE main page

How test will be performed: Tester will check stay logged in checkbox go to main page, leave REVITALIZE, reopen REVITALIZE and check whether same data from main page is the same from the last time tester opened main page

8. FR8

Control: Manual

Initial State: Login page is displayed with sign up button

Input: Click sign up button

Output: Loads and displays sign up page

Test Case Derivation: User can click sign up button which loads and displays sign up page

How test will be performed: Tester will click on sign up button and checks if sign up page is displayed

9. FR9

Control: Manual

Initial State: Login page is displayed with inputted information in username and password text-boxes

Input: Click login button

Output: if failure state, display an invalid password or username banner, else if success state, load and display main page

Test Case Derivation: User clicks login button and request is made based on username and password, and will proceed to main page only if username and password are valid

How test will be performed: Tester will click on login button and test for scenarios when login should be successful and when login should fail

5.1.2 Area of Testing2

...

5.2 Tests for Nonfunctional Requirements

[The nonfunctional requirements for accuracy will likely just reference the appropriate functional tests from above. The test cases should mention reporting the relative error for these tests. —SS]

[Tests related to usability could include conducting a usability test and survey. —SS]

5.2.1 Area of Testing1

Title for Test

1. test-id1

Type:

Initial State:

Input/Condition:

Output/Result:

How test will be performed:

2. test-id2

Type: Functional, Dynamic, Manual, Static etc.

Initial State:

Input:

Output:

How test will be performed:

5.2.2 Area of Testing2

...

5.3 Traceability Between Test Cases and Requirements

[Provide a table that shows which test cases are supporting which requirements. —SS]

6 Unit Test Description

[Reference your MIS and explain your overall philosophy for test case selection. —SS] [This section should not be filled in until after the MIS has been completed. —SS]

6.1 Unit Testing Scope

[What modules are outside of the scope. If there are modules that are developed by someone else, then you would say here if you aren't planning on verifying them. There may also be modules that are part of your software, but have a lower priority for verification than others. If this is the case, explain your rationale for the ranking of module importance. —SS]

6.2 Tests for Functional Requirements

[Most of the verification will be through automated unit testing. If appropriate specific modules can be verified by a non-testing based technique. That can also be documented in this section. —SS]

6.2.1 Module 1

[Include a blurb here to explain why the subsections below cover the module. References to the MIS would be good. You will want tests from a black box perspective and from a white box perspective. Explain to the reader how the tests were selected. —SS]

1. test-id1

Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

Initial State:

Input:

Output: [The expected result for the given inputs —SS]

Test Case Derivation: [Justify the expected value given in the Output field —SS]

How test will be performed:

2. test-id2

Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

Initial State:

Input:

Output: [The expected result for the given inputs —SS]

Test Case Derivation: [Justify the expected value given in the Output field —SS]

How test will be performed:

3. ...

6.2.2 Module 2

...

6.3 Tests for Nonfunctional Requirements

[If there is a module that needs to be independently assessed for performance, those test cases can go here. In some projects, planning for nonfunctional tests of units will not be that relevant. —SS]

[These tests may involve collecting performance data from previously mentioned functional tests. —SS]

6.3.1 Module ?

1. test-id1

Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

Initial State:

Input/Condition:

Output/Result:

How test will be performed:

2. test-id2

Type: Functional, Dynamic, Manual, Static etc.

Initial State:

Input:

Output:

How test will be performed:

6.3.2 Module ?

...

6.4 Traceability Between Test Cases and Modules

[Provide evidence that all of the modules have been considered. —SS]

References

Author Author. System requirements specification. <https://github.com/...>, 2019.

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

7.2 Usability Survey Questions?

[This is a section that would be appropriate for some projects. —SS]