SE 3XA3: Test Plan
Wordle 2.0

Team 8, Goufs
Richard Fan, fanr13
Noel Zacharia, zacharin
Biranugan Pirabaharan, pirabahb

March 11, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
Mar 1, 2022	1.0	Initial Document
Mar 3, 2022	1.1	Finished section 1 and 2
Mar 5, 2022	1.2	Tests for functional requirements
Mar 9, 2022	1.3	Tests for non functional requirements
Mar 10, 2022	1.4	Traceability Matrices added
Mar 11, 2022	1.5	Fixed grammar mistakes

1 General Information

1.1 Purpose

This document describes the tests, methods and tools that will be used to verify the functional and non-functional requirements for Wordle 2.0. These methods serve as a guide to test the software once it has been fully implemented. The purpose of these tests is to discover and correct any potential errors that may have occurred during implementation and ensure adherence to the Software Requirements Specification.

1.2 Scope

The test cases described within this document cover all requirements, both non-functional and functional, that were specified within the Software Requirements Specification. Tests will be conducted as development progresses and this document will be updated accordingly.

1.3 Acronyms, Abbreviations, and Symbols

Table 2: **Table of Abbreviations**

Abbreviation	**Definition**
CSS	Cascading Style Sheets
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JS	Javascript

1.4 Overview of Document

The document contains a description of the software, the tools and methods used to test the software, and the team that will be testing it. It includes all the test cases that will be used to verify the software's adherence to the all SRS requirements. Also, a comparison to the original Wordle game is provided.

Table 3: **Table of Definitions**

Term	**Definition**
Cascading Style Sheets	The stylesheet language used to style the project
Javascript[JS]	The programming language used to provide interactivity in the project
Game Board	A collection of boxes on which the letters will be displayed.
Graphical User Interface	A representation of the program allowing for user interaction.
HTML	The markup language used to structure the project
Keyboard	The area on the screen where the player selects individual letters.
Mocha	One of the testing frameworks used.
Not Wordle	The original game on which this project is based.
Player	The individual who is playing the game.
Player Statistics	A collection of values showing the win/loss rate and number of guesses.
Selenium	The UI testing framework used.
Software Requirements Specification	A document that describes a software system's performance and functionality.
Tester	A person who is testing the program by way of using the program.
Typescript	The programming language used in the original Not Wordle game.

2 Plan

2.1 Software Description

Wordle 2.0 is the reimplementation of the game Wordle. It is built using JavaScript, CSS, and HTML.

2.2 Test Team

All members of Group-8 are responsible for writing and executing tests:

1. Richard Fan
2. Biranugan Pirabaharan
3. Noel Zacharia

In addition to this, once the product is in the final stages of development, we would seek user feedback by recruiting volunteers.

2.3 Automated Testing Approach

As our project is a GUI-based game, automated testing will be a minor aspect of our testing framework. We plan on testing via obtaining user feedback and verification of different game scenarios. However, we will be writing automated unit tests for different components of the game using the Selenium Testing Framework. For unit tests concerning internal functions, we will be using Mocha.

2.4 Testing Tools

Mocha is a JavaScript testing framework that runs both on Node.js and the browser. Mocha will allow for unit tests to be conducted on internal functions within the game. The main testing tool we will be using is Selenium. Selenium is an open-source testing framework for web applications. Selenium allows for automated testing of the UI components and thus allows us to verify the rendering of page elements. Python will be used to write Selenium test scripts.

2.5 Testing Schedule

The testing schedule is described in the Gantt chart.

3 System Test Description

3.1 Tests for Functional Requirements

3.1.1 User Interface Tests

1. FR-UI1

   Type: Functional, Manual

   Initial State: Empty web browser page.

   Input: A start game event.

   Output: An on screen keyboard with a QWERTY layout, containing enter and delete keys.

   How test will be performed: A tester will open the game within a web browser and check that an on screen keyboard of QWERTY layout is displayed.

2. FR-UI2

   Type: Functional, Manual

   Initial State: An blank web browser page.

   Input: A start game event.

   Output: A game board of with 6 columns and 5 rows.

   How test will be performed: A tester will open the game within a web browser and check that a game board with 5 rows and 6 columns is displayed.

3. FR-UI3

   Type: Functional, Automated, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

Input: The theme button is pressed.

Output: The theme of the game changes.

How test will be performed: An automated script will click on the theme button and check if the styling has changed.

4. FR-UI4

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: The rules button is pressed.

   Output: A message box is displayed containing the instructions of the game.

   How test will be performed: A tester will click on the instructions button and check that a message box containing the instructions is displayed.

5. FR-UI5

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: The statistics button is pressed.

   Output: A message box is displayed containing all of the player statistics.

   How test will be performed: A tester will click on the statistics button and check that a message box containing the player statistics is displayed.

6. FR-UI6

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: An end game event.

   Output: A message box is displayed containing the number of games played.

How test will be performed: A tester will do a playthrough of the game. At the end they will check that a message box containing the total number of games is displayed.

7. FR-UI7

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: An end game event.

   Output: A message box is displayed containing the win/loss ratio.

   How test will be performed: A tester will do a playthrough of the game. At the end, they will check that a message box containing the player's win/loss ratio is displayed.

8. FR-UI8

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: An end game event.

   Output: A message box is displayed containing the current streak of correct guesses.

   How test will be performed: A tester will do a playthrough of the game. At the end, they will check that a message box containing the player's current streak of correct guesses is displayed.

9. FR-UI9

   Type: Functional, Manual, Dynamic

   Initial State: A web browser with the Wordle 2.0 game loaded.

   Input: An end game event.

   Output: A message box is displayed containing the best streak of correct guesses.

   How test will be performed: A tester will do a playthrough of the game. At the end they will check that a message box containing the player's best streak of correct guesses is displayed.

10. FR-UI10

    Type: Functional, Manual, Dynamic

    Initial State: A web browser with the Wordle 2.0 game loaded.

    Input: An end game event.

    Output: A message box is displayed, which contains the player's guess distribution.

    How test will be performed: A tester will do a playthrough of the game. At the end, they will check if a message box containing the player's guess distribution is displayed.

3.1.2 Gameplay Tests

1. FR-GP1

    Type: Functional, Manual

    Initial State: The Wordle 2.0 site has been initialized.

    Input: Tester clicks the share option.

    Output: The clipboard stores a copy of the results.

    How test will be performed: A tester will play the game and at the end of the game, will click the share option, paste the results into a word document and check whether the desired image has been pasted.

2. FR-GP2

    Type: Functional, Manual

    Initial State: The Wordle 2.0 site has been initialized.

    Input: Tester enters a word.

    Output: The gameboard allows the user to enter the word.

    How test will be performed: A tester will play the game and enter a 5-letter word and hit submit. The tester will observe if the gameboard accepts the word or not.

3. FR-GP3

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters a word with a letter in the correct spot.

   Output: The gameboard highlights the correct letter tile in green.

   How test will be performed: A tester will play the game and enter a 5-letter word with one letter in the correct spot and will observe whether or not the gameboard highlights that tile in green.

4. FR-GP4

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters a word with a letter in the correct spot.

   Output: The gameboard highlights the correct letter tile in green.

   How test will be performed: A tester will play the game and enter a 5-letter word with one letter in the correct spot and observes whether the gameboard highlights that tile in green.

5. FR-GP5

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters a word with a letter in the wrong spot.

   Output: The gameboard does not highlight the letter tile.

   How test will be performed: A tester will play the game and enter a 5-letter word with letters in the wrong spot and observes whether the gameboard does not highlight that tile.

6. FR-GP6

   Type: Functional, Manual

Initial State: The Wordle 2.0 site has been initialized.

Input: Tester enters an invalid 5-letter word.

Output: The game alerts the user about the invalid word.

How test will be performed: A tester will play the game and enter an invalid word. The game should alert the user about this.

7. FR-GP7

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters an invalid word with less than 5 letters.

   Output: The game alerts the user about the invalid word.

   How test will be performed: A tester will play the game and enter an invalid word with less than 5 letters. The game should alert the user about this.

8. FR-GP8

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters a word with some correct letters.

   Output: The game keyboard updates itself to reflect the accuracy of the new guess.

   How test will be performed: A tester will play the game and enter a word. The keyboard will update itself to reflect the letters which were within the word and those that were not.

9. FR-GP9

   Type: Functional, Manual

   Initial State: The Wordle 2.0 site has been initialized.

   Input: Tester enters a word and tries to modify that word after submission.

Output: The game does not allow any modifications after submission.

How test will be performed: A tester will play the game, enter a word and try to modify that word after submission, however, the game must ensure that this does not happen.

10. FR-GP10

    Type: Functional, Manual

    Initial State: The Wordle 2.0 site has been initialized.

    Input: Tester sees an option to change the game word length setting.

    Output: The game provides the user option to change the word length.

    How test will be performed: A tester will see the option to change word length in the game and the game allows them to alternate between different modes.

11. FR-GP11

    Type: Functional, Manual

    Initial State: The Wordle 2.0 site has been initialized.

    Input: Tester clicks the change word-length button.

    Output: The game changes the word length setting according to player inputs.

    How test will be performed: A tester will change the game length and enter words of corresponding lengths to ensure that the game functions as desired in all settings.

12. FR-GP12

    Type: Functional, Manual

    Initial State: The Wordle 2.0 site has been initialized.

    Input: Tester clicks the reset button.

    Output: The game resets for another round of play.

    How test will be performed: A tester will click the reset button and the game should allow them to play another round with a new word.

3.2 Tests for Nonfunctional Requirements

3.2.1 Look and Feel Testing

1. NFR-LF1

   Type: Functional, Dynamic, Manual

   Initial State: Game loaded with no guesses inputed.

   Input: **Tester** plays the game by guessing words.

   Output: Survey results

   How test will be performed: The responses of the **testers** to question 2 and 3 of the Usability Survey will determine the ease of use and understandability of the game.

2. NFR-LF2

   Type: Functional, Dynamic, Manual

   Initial State: Game loaded with no guesses made.

   Input: **Tester** plays the game by guessing words.

   Output: Survey results

   How test will be performed: **Testers** will play the game and the original **Wordle 2.0** and answer questions 1 and 4 from the Usability Survey.

3.2.2 Usability and Humanity Testing

1. NRF-UH1

   Type: Functional, Dynamic, Manual

   Initial State: Game loaded with no guesses made.

   Input: **Tester** plays the game by guessing words.

   Output: Survey results

   How test will be performed: Children at or above the age of MIN_AGE will play the game and fill out a survey that evaluates their understanding.

2. NRF-UH2

   Type: Functional, Dynamic, Manual

   Initial State: Game loaded with no guesses made.

   Input: **Tester** plays the game by guessing words.

   Output: Survey results

   How test will be performed: **Testers** will play the game and will fill out a survey that evaluates how easy they found the game to be playable with one hand

3. NRF-UH4

   Type: Functional, Dynamic, Manual

   Initial State: Game loaded with no guesses made.

   Input: **Tester** plays the game by guessing words.

   Output: Survey Results

   How test will be performed: The responses of the **testers** to question 5 of the Usability Survey will determine the difficulty of the game.

3.2.3 Performance Testing

1. NRF-P1

   Type: Functional, Automatic

   Initial State: Game loaded with no guesses made.

   Input: A guess will be inputted.

   Output: Survey Results

   How test will be performed: An automated unit test will check the time it takes for the page elements to be added to the element tree and be displayed. It must be less than MIN_TIME, over multiple executions and varying devices.

2. NRF-P2

   Type: Functional, Automatic

   Initial State: Game is to be initialized.

   Input: The site will be reloaded

   Output: Survey Results

   How test will be performed: An automated unit test will reload the page to check the time it takes for the page elements to be loaded to the element tree and be displayed. It must be less than MIN_TIME, over multiple executions and varying devices.

3.3 Traceability Between Test Cases and Requirements

Table 4: **Traceability Matrix for UI Requirements**

		Requirements									
		FR1	FR2	FR3	FR4	FR5	FR6	FR7	FR8	FR9	FR10
Test Cases	FR-UI1	X									
	FR-UI2		X								
	FR-UI3			X							
	FR-UI4				X						
	FR-UI5					X					
	FR-UI6						X				
	FR-UI7							X			
	FR-UI8								X		
	FR-UI9									X	
	FR-UI10										X

Table 5: **Traceability Matrix for Gameplay Requirements**

		Requirements											
		FR11	FR12	FR13	FR14	FR15	FR16	FR17	FR18	FR19	FR20	FR21	FR22
Test Cases	FR-GP1	X											
	FR-GP2		X										
	FR-GP3			X									
	FR-GP4				X								
	FR-GP5					X							
	FR-GP6						X						
	FR-GP7							X					
	FR-GP8								X				
	FR-GP9									X			
	FR-GP10										X		
	FR-GP11											X	
	FR-GP12												X

Table 6: **Traceability Matrix for Non Functional Requirements**

		Requirements						
		LF1	LF2	UH1	UH2	UH4	P1	P2
Test Cases	NFR-LF1	X						
	NFR-LF2		X					
	NFR-UH1			X				
	NFR-UH2				X			
	NFR-UH4					X		
	NFR-P1						X	
	NFR-P2							X

4 Tests for Proof of Concept

4.1 Gameplay Testing

1. POC-GP-1

   Type: Functional, Manual

   Initial State: Initialization of the Wordle board and keyboard

   Input: The webpage is reloaded

   Output: A blank board and default coloured keyboard are displayed.

   How test will be performed: The tester will load a new game and check that the game's guess board and on screen keyboard appear.

2. POC-GP-2

   Type: Functional, Manual

   Initial State: Game should be initialized

   Input: **Tester** submits a word with less than 5 letters.

   Output: Player is alerted the word is too short.

   How test will be performed: The **tester** submits a word with less than 5 letters, and checks to see if they are alerted in response.

3. POC-GP-3

   Type: Functional, Manual

   Initial State: Game should be initialized

   Input: **Tester** submits a fake word.

   Output: Player is alerted the word is invalid.

   How test will be performed: The **tester** submits a gibberish word, and checks to see if they are alerted in response.

4. POC-GP-4

   Type: Functional, Manual

   Initial State: Game should be initialized

   Input: **Tester** guesses various word.

   Output: The game board and on-screen keyboard updates.

How test will be performed: The **tester** submits various words, through both input methods, and checks to see if the game board and on-screen keyboard updates with the appropriate colours.

5. POC-GP-5

   Type: Functional, Manual

   Initial State: Game should be initialized

   Input: The target word is submitted.

   Output: The game no longer accepts any new guesses.

   How test will be performed: The test automatically guesses the target word, then a **tester** checks both the physical and on-screen keyboards.

6. POC-GP-6

   Type: Functional, Manual

   Initial State: Game should be initialized

   Input: The dark mode switch is clicked.

   Output: The game switches themes.

   How test will be performed: The **tester** clicks the dark mode switch, and checks to see if all the elements switch to their dark mode colours. The tester should click the switch again to ensure it can go both ways.

5 Comparison to Existing Implementation

At the time of writing this document, Wordle and Wordle 2.0 both have mirrored core functionality as both provide an interface to input 5-letter words and give information on the accuracy of the input letters. Things that remain to be implemented are additional menu options for player statistics, dynamic word levels and improved graphics.

6 Unit Testing Plan

6.1 Unit testing of internal functions

We will be using Mocha and the built-in assertion library provided by Node.js to test internal functions. We will write a test file consisting of different

functions that correspond to tests for each module. We will be using assertion statements, which take predetermined parameters and compares the method output with the expected output. If the assertion statement evaluates to true, we can say that this test has passed successfully and is correct for this case. To ensure coverage of the input domain, we will choose extreme/boundary values as our predetermined parameters. The Selenium suite will be used also be used to test internal functions that can not be tested using Mocha. We will be using a page object design pattern for our tests. Selenium will be used to get HTML elements from the game and pass inputs to them. Assert statements will then be used to compare the system output with the expected output. This may be in the form of locating new elements on the game page etc. It is expected that unit testing will be able to cover $UNIT\_TEST\_COVERAGE$ percent of the code. The rest of the code will be verified using manual testing.

6.2 Unit testing of output files

There are no output files created from **Wordle 2.0**, therefore there are no unit tests for this section.

7 Appendix

7.1 Symbolic Parameters

$MIN\_AGE = 10$
$MIN\_TIME = 0.25$
$UNIT\_TEST\_COVERAGE = 40$

7.2 Usability Survey Questions?

Tester feedback is an important component of testing and their experience playing Wordle should be taken into account. The following are some survey questions we could ask:

1. How would you compare your experience playing Wordle 2.0 to the official version of Wordle?
2. On a scale from 1-10, how intuitive were the controls?
3. On a scale from 1-10, how easy was it to enter inputs within the game?
4. Would you consider the UI to be visually appealing?
5. How long did it take you to figure out the instructions on how to play this game?