Software Development Project - 1DV600 - Test documentation

Claes Weyde
e-mail: cw222av@student.lnu.se
github: https://github.com/Cosbus/cw222av_1dv600_

March 7, 2019

Contents

1 Test Plan

1.1 Objectives

The objective of this testing phase is to snuff out as many problems with the application as possible. The testing is aimed at making sure the critical system functions are operational. As this is primarily a game, a large focus is put on the gaming part to make sure that it is working properly. However, since the words used in the game are of great importance to minimize repetitiveness quite a large focus is put on the functionality regarding adding, deleting and updating the words.

1.1.1 What to test?

As was described above the main testing will be performed on the gaming part of the application. Manual test will be performed to determine that all the critical functionality works with regards to game play, specifically UC2 - Play game. The main scenario as well as some alternative scenarios will be tested.

- Choosing a game difficulty.
- Providing a letter successfully and unsuccessfully.
- Quitting the game.
- Finishing a word.

Some attention will also be put on managing words, i.e. UC 5, 6 and 7. All of the manual dynamic tests are to be performed after the code for the use cases have been implemented. For a full breakdown of the manual testing please see section 2.2.

Automated unit tests will also be written to test some of the methods provided in the different objects of the application.

1.2 To-do

- Plan work
  - Is required functionality missing?

  - Should new functionality be implemented?
  - Prioritize what needs to be done.
- Update project document.
- Possible: Implement new features.
  - Prepare UML for new features.
  - Implement new features.
  - Test new features as well as whole game.
- Complete and hand in project.

1.3 Time estimates and Time log

Task	Estimate [hrs]	Actual time
Plan work	2	–
Document update	1	–
Possible implementation	10-15	–
Finish up and hand in	2	–
Total time	20	–

2 Manual test cases

2.1 Manual tests of Play Game main scenario

2.1.1 TC 2.2 - Successfully providing a correct letter

- Name: Successfully providing a correct letter.
- Test case ref: TC 2.2
- Use case ref: UC 2
- Description: The test case tests how the system responds when the user provides a letter which is correct, and the word is not completely solved.

Input

1. Precondition: The user has started the game.
2. Test case TC 2.1. choose "play game" from main menu.
3. Input a correct letter.
4. Press enter.

Output

- The system prints "Correct!" on the screen.
- The system prints the number of letters remaining on the screen.
- The system prints the number of tries remaining on the screen.
- The system prompts the user to enter a new letter by printing "Please provide a letter:" on the screen.

Test successful ☐
Test not successful ☐

Comments

2.1.2 TC 2.3 - Providing an incorrect letter

- Name: Providing an incorrect letter.
- Test case ref: TC 2.3
- Use case ref: UC 2
- Description: The test case tests how the system responds when the user provides a letter which is incorrect, and the user has more tries.

Input

1. Precondition: The user has started the game.
2. Test case TC 2.1. choose "play game" from main menu.
3. Input an incorrect letter.
4. Press enter.

Output

- The system prints "Unfortunately wrong." on the screen.
- The system prints the number of letters remaining on the screen.
- The system prints the number of tries remaining on the screen.
- The system prompts the user to enter a new letter by printing "Please provide a letter:" on the screen.

Test successful ☐
Test not successful ☐

Comments

2.1.3 TC 2.6 - Returning from play to main menu

- Name: Returning from play to main menu.
- Test case ref: TC 2.6
- Use case ref: UC 2
- Description: The test case tests how the system responds when the user has finished a word (either by finding the word or having exhausted his/hers amount of tries) and chooses to return to the main menu.

Input

1. Precondition: The user has finished a word and the system has provided the user with the choice of returning to main menu, playing a new word or quitting the game altogether.
2. Finish a word.
3. Input "m" to return to main menu.
4. Press enter.

Output

- The system shows the main menu on the screen.

Test successful ☐
Test not successful ☐

Comments

2.1.4 TC 2.a* - Quitting the game while playing

- Name: Quitting the game while playing.
- Test case ref: TC 2.a*
- Use case ref: UC 2
- Description: The test case tests how the system responds when the user enters the code for quitting the game while playing.

Input

1. Precondition: The user is playing the game.
2. Input "q-t" and press enter.
3. The system asks the user if he/she really wants to quit.
4. Input "y" and press enter.

Output

- The system quits the application entirely.

Test successful ☐
Test not successful ☐

Comments

2.1.5 TC 5.3 - Inputting a new word

- Name: Inputting a new word.
- Test case ref: TC 5.3
- Use case ref: UC 5
- Description: The test case tests how the system responds when the user enters a new word to save to the word list.

Input

1. Precondition: The user has moved to the manage words menu and chosen to add a new word. The system prompts the user to input the word by printing "Input a word:" on the screen.
2. Input a word and press enter.

Output

- The system prints "Make a choice: (use arrow keys)".
- The system prints the three difficulty levels "easy", "moderate" and "hard" which the user can move between.

Test successful ☐
Test not successful ☐

Comments

2.1.6 TC 5.5 - Choosing a difficulty for an input word

- Name: Choosing a difficulty for an input word.
- Test case ref: TC 5.5
- Use case ref: UC 5
- Description: The test case tests how the system responds when the user enters a difficulty for an input word.

Input

1. Precondition: The user has input a word and pressed enter.
2. TC 5.3. Choose a difficulty presented by the system.
3. Press enter.

Output

- The system prints "Successfully saved" in green text.
- The system prints "Add another word? [y/n]" and waits for the user to respond.

Test successful ☐
Test not successful ☐

Comments

2.1.7 TC 6.2 - Choosing to remove a word

- Name: Choosing to remove a word.
- Test case ref: TC 6.2
- Use case ref: UC 6
- Description: The test case tests how the system responds when the user chooses to remove a word.

Input

1. Precondition: The user has moved to the manage words menu and the system has printed the menu on the screen.
2. Choose to remove a word by moving the cursor to the "delete word" choice using the arrow keys.
3. Press enter.

Output

- The system prints "Remove word. What difficulty are you interested in" on the screen.
- The system prints "Make a choice: (use arrow keys)" on the screen.
- The system prints the choices "easy", "moderate" and "hard" on the screen and waits for the user choice.

Test successful ☐
Test not successful ☐

Comments

2.1.8 TC 6.3.1 - Choosing a difficulty for the word to remove

- Name: Choosing a difficulty for the word to remove.
- Test case ref: TC 6.3.1
- Use case ref: UC 6
- Description: The test case tests how the system responds when the user chooses a difficulty for the word to be removed.

Input

1. Precondition: The user has chosen to remove a word (TC 6.2).
2. Choose a difficulty and press enter.

Output

- The system prints all the words on file for the given difficulty on the screen and waits for the user choice. The user can move between the words using the arrow keys to choose.

Test successful ☐
Test not successful ☐

Comments

2.1.9 TC 6.3.2 - Choosing a word to remove from the list

- Name: Choosing a word to remove from the list.
- Test case ref: TC 6.3.2
- Use case ref: UC 6

- Description: The test case tests how the system responds when the user chooses a word from the list to remove.

Input

1. Precondition: The user has chosen a difficulty for the word to remove (TC 6.3.1).
2. Choose a word from the list by using the arrow keys.
3. Press enter.

Output

- The system prints "Are you sure you want to remove "[thechoosen-word]" [y/n]?" and awaits the users answer.

Test successful ☐
Test not successful ☐

Comments

2.1.10 TC 6.3.2a - Confirming removal of a chosen word

- Name: Confirming removal of a chosen word.
- Test case ref: TC 6.3.2a
- Use case ref: UC 6
- Description: The test case tests how the system responds when the user confirms removal of the chosen word.

Input

1. Precondition: The user has chosen a word to remove from the list (TC 6.3.2).
2. Write "y".
3. Press enter.

Output

- The system prints "[word] successfully removed!" in green text.
- The system prints "Delete another word? [y/n]" and awaits the user response.

Test successful ☐
Test not successful ☐

Comments

2.1.11 TC 6.3.2b - Not confirming to remove a chosen word

- Name: Not confirming to remove a chosen word.
- Test case ref: TC 6.3.2b
- Use case ref: UC 6
- Description: The test case tests how the system responds when the user chooses to not confirm to remove a word chosen for removal.

Input

1. Precondition: The user has chosen a word to remove from the list (TC 6.3.2).

2. Write "n".
3. Press enter.

Output

- The system prints "Maybe a wise choice?".
- The system prints "Do you want to remove another word [y/n]? and awaits the users answer.

Test successful ☐
Test not successful ☐

Comments

2.1.12 TC 7.3 - Choosing a word to update

- Name: Choosing a word to update.
- Test case ref: TC 7.3
- Use case ref: UC 7
- Description: The test case tests how the system responds when the user chooses a word to update.

Input

1. Precondition: The user has chosen to update a word from the manage word menu and chosen which difficulty the word should belong to (TC 7.2).
2. Choose a word by using the arrow keys.

3. Press enter.

Output

- The system prints "What word do you want to update "[word]" to?".
- The system awaits the user choice.

Test successful ☐
Test not successful ☐

Comments

2.1.13 TC 7.5 - Choosing a word to update to

- Name: Choosing a word to update to.
- Test case ref: TC 7.5
- Use case ref: UC 7
- Description: The test case tests how the system responds when the user inputs a word to update to.

Input

1. Precondition: The user has chosen a word to update from the list of words (TC 7.2).
2. Write the word to update to.
3. Press enter.

Output

- The system prints "Are you sure you want to update "[oldword]" to "[newword]" [y/n]?" and awaits the user answer.

Test successful ☐
Test not successful ☐

Comments

2.1.14 TC 7.6 - Answering with wrong letter to [y/n] question

- Name: Answering with wrong letter to [y/n] question.
- Test case ref: TC 7.6
- Use case ref: UC 7
- Description: The test case tests how the system responds when the user answers with an un-allowed letter to a yes/no question when updating words.

Input

1. Precondition: The user has entered a word to update to (TC 7.5).
2. Write any letter or number which differs from "y" or "n".
3. Press enter.

Output

- The system prints "Please answer "y" or "n"!".
- The system prints "Are you sure you want to update "[oldword]" to "[newword]" [y/n]?" and awaits the user answer.

Test successful ☐
Test not successful ☐

Comments

2.2 Test report for automated test

Test	**UC 2**	**UC 4**	**UC 5**	**UC 6**	**UC7**
TC 2.2	1	0	0	0	0
TC 2.3	1	0	0	0	0
TC 2.6	1	0	0	0	0
TC 2.a*	1	0	0	0	0
TC 5.3	0	0	1	0	0
TC 5.5	0	0	1	0	0
TC 6.2	0	0	0	1	0
TC 6.3.1	0	0	0	1	0
TC 6.3.2	0	0	0	1	0
TC 6.3.2a	0	0	0	1	0
TC 6.3.2b	0	0	0	1	0
TC 7.3	0	0	0	0	1
TC 7.5	0	0	0	0	1
TC 7.6	0	0	0	0	1
Coverage and success	4/OK	0	2/OK	4/OK	3/OK