SE 3XA3: Test Report
SpaceShooter

Team #105, Space Shooter
Dananjay Prabaharan - prabahad
Nishanth Raveendran - raveendn
Hongzhao Tan - tanh10

April 7, 2020

Contents

1 Functional Requirements Evaluation **1**
1.1 Keys . . . 1
1.2 Movement and Shooting . . . 3
1.3 Collision . . . 5
1.4 Death . . . 6

2 Nonfunctional Requirements Evaluation **7**
2.1 Look and Feel . . . 7
2.2 Usability and Humanity . . . 7
2.3 Performance . . . 8
2.4 Maintainability . . . 9
2.5 Cultural Requirement . . . 10

3 Unit Testing **10**

4 Changes Due to Testing **11**

5 Automated Testing **11**

6 Trace to Requirements **12**

7 Trace to Modules **14**

8 Code Coverage Metrics **15**

List of Tables

1 **Revision History** . . . ii
2 Trace Between Tests and Requirements . . . 13
3 Trace Between Tests and Modules . . . 14

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
April 2, 2020	1.0	Initial Test Report
April 6, 2020	1.1	Rev1 Update

1 Functional Requirements Evaluation

1.1 Keys

Main Menu

1. FS-MM-1

   Initial State: Screen is initially at the Main Menu Screen displaying three options: Play, Instructions, and Quit.

   Input: User clicks 'Play' Key

   Expected Output: Program redirects to the 'Playing Game State' where the user could play the game.

   Result: PASS

2. FS-MM-2

   Initial State: Screen is initially at the Main Menu Screen displaying three options: Play, Instructions, and Quit.

   Input: User clicks 'Instructions' Key

   Expected Output: Program directs to the 'Pre-Game State' where the user could read the instructions to the game.

   Result: PASS

3. FS-MM-3

   Initial State: Screen is initially at the Main Menu Screen displaying three options: Play, Instructions, and Quit.

   Input: User clicks 'Quit' Key

   Expected Output: Program quits and disappears from the user's screen.

   Result: PASS

4. FS-MM-4

   Initial State: Screen is initially at the Instruction Screen.

Input: User clicks ’Back’ Key

Expected Output: Program directs to the ’Pre-Game State’ where the user is at the Main Menu Screen.

Result: PASS

5. FS-MM-5

   Initial State: Screen is initially at the Main Menu Screen displaying three options: Play, Instructions, and Quit.

   Input: User clicks ’Play’ Key and immediately hits the ’Instruction’ Key

   Expected Output: Program displays error stating ”Instructions Page cannot be displayed while launching the game”

   Result: FAIL

Paused

1. FS-P-1

   Initial State: Screen is initially at the ’Playing Game State’.

   Input: User clicks ’Pause’ Key

   Expected Output: Program redirects to the ’Paused State’ where the user is at the paused game state.

   Result: PASS

2. FS-P-2

   Initial State: Screen is initially at the ’Paused State’.

   Input: User clicks ’Resume’ Key

   Expected Output: Program directs to the ’Playing Game State’ where the user resumes from the left off position and state of the game.

   Result: PASS

3. FS-P-3

   Initial State: Screen is initially at the ’Paused State’.

   Input: User clicks ’Pause’ Key while in ’Paused State’

   Expected Output: Program remains on ’Paused State’

   Result: PASS

4. FS-P-4

   Initial State: Screen is initially at the ’Paused State’.

   Input: User clicks ’Resume’ Key and while the program attempts to resume the game, the user immediately clicks the ’Pause’ Key

   Expected Output: Program ignores ’Resume’ Key request and redirects back to the ’Paused State’

   Result: FAILS

5. FS-P-5

   Initial State: Screen is initially at the ’Paused State’.

   Input: User clicks ’Quit’ Key while in ’Paused State’

   Expected Output: Program quits and disappears from the user’s screen.

   Result: PASS

1.2 Movement and Shooting

Spaceship Movement Operations

1. FS-M-1

   Initial State: Screen is initially at the ’Playing Game’ State.

   Input: User presses one directional control button (right/left/up/down).

   Expected Output: Spaceship immediately moves to the respective direction based on what the user presses.

   Result: PASS

2. FS-M-2

   Initial State: Screen is initially at the 'Playing Game' State.

   Input: User presses one directional key and immediately presses another directional key while pressing the original key too (ex: press 'right' followed by 'left').

   Expected Output: Spaceship should move in the direction of the second directional input and ignore the first directional input when pressing both at the same time.

   Result: FAILS

3. FS-M-3

   Initial State: Screen is initially at the 'Playing Game' State.

   Input: User presses two directional keys at the same time (ex: press 'right' and 'left' together).

   Expected Output: Spaceship shall not move and maintain its position of where it was left off.

   Result: PASS

Spaceship Shooting Operations

1. FS-S-1

   Initial State: Screen is initially at the 'Playing Game' State.

   Input: User presses shooting button at a upcoming obstacle.

   Expected Output: An animation would proceed where a bullet comes from the spaceship and hits the respective obstacle. The obstacle would receive damage based on the number of bullets that hits it. If it receives its max damage, it explodes.

   Result: PASS

1.3 Collision

Collision Detection

1. FS-CD-1

   Initial State: Screen is initially at the 'Playing Game State' where obstacles (meteors and aliens) are approaching the spaceship.

   Input: Obstacle collides with the spaceship.

   Expected Output: Spaceship loses health according to size/power of the obstacle.

   Result: PASS

2. FS-CD-2

   Initial State: Screen is initially at the 'Playing Game State' where a "Faster Shooting" power-up is approaching the spaceship.

   Input: "Faster Shooting" power-up collides with the spaceship.

   Expected Output: The spaceship shoots bullets twice the speed for 5 seconds.

   Result: PASS

3. FS-CD-3

   Initial State: Screen is initially at the 'Playing Game State' where a "Double Shooting" power-up is approaching the spaceship.

   Input: "Double Shooting" power-up collides with the spaceship.

   Expected Output: The spaceship shoots two bullets at a time for 5 seconds.

   Result: PASS

4. FS-CD-4

   Initial State: Screen is initially at the 'Playing Game State' where a "Shield" power-up is approaching the spaceship.

Input: ”Shield” power-up collides with the spaceship.

Expected Output: The spaceship is gains a boost of 30% of health.

Result: PASS

5. FS-CD-5

   Initial State: Screen is initially at the ’Playing Game State’ where a ”Missile” power-up is approaching the spaceship.

   Input: ”Missile” power-up collides with the spaceship.

   Expected Output: The spaceship begins to start shooting missiles for 5 seconds.

   Result: PASS

6. FS-CD-6

   Initial State: Screen is initially at the ’Playing Game State’ where the spaceship is already in effect from a previous power-up.

   Input: A power-up collides with the spaceship while it still has effect from a previous power-up.

   Expected Output: The spaceship stacks the power-ups and uses both at the same time.

   Result: FAIL

1.4 Death

1. FS-D-1

   Initial State: Screen is initially at the ’Playing Game State’ where obstacles are approaching the spaceship. Spaceship is close to losing all health.

   Input: Obstacle collides with the spaceship, causing the health bar to be **at 0**.

   Expected Output: Spaceship’s destroyed and the game is redirected to the ’Pre-Game State’ with the Main Menu.

Result: PASS

2. FS-D-2

   Initial State: Screen is initially at the 'Playing Game State' where obstacles are approaching the spaceship. Spaceship is close to losing all health.

   Input: Obstacle collides with the spaceship, causing the health bar to be **below 0**.

   Expected Output: Spaceship's destroyed and the game is redirected to the 'Pre-Game State' with the Main Menu.

   Result: PASS

2 Nonfunctional Requirements Evaluation

2.1 Look and Feel

1. NFS-LF-1

   Description: To test the look and feel of the game, the each of the development team members have been playing the game for 5 times from start playing the game to "Death" of the space shooter then answered the questions specified in the test plan.

   Initial State: The system is initially at the "Pre-Game State"

   Input: Development Team Members

   Expected Output: The answers from the testers to the set of questions specified in the test plan about look and feel of the game are all positive.

   Result: PASS

2.2 Usability and Humanity

1. NFS-UH-1

Description: To test the usability of the game, the development team invited 4 outside people who had very little knowleadge about the two-dimensional platform type games to each play the game for 10 times from the start of the game to "Death" of the space shooter, with what keys on the keyboard could be used to control the space shooter being acknowledged to these testers previously. After playing the game, the testers were asked questions about what each of the keys which the development team had told them can do in terms of controlling the space shooter.

Initial State: The system is initially at the "Pre-Game State"

Input: 4 Invited Outside People

Expected Output: At least the answers from 3 out of the 4 testers to the set of questions about controlling the space shooter are all correct.

Result: PASS

2.3 Performance

1. NFS-P-1

   Description: To test the response speed of the system, one of the members of the development team manually sent all possible requests into the system under main-menu scene, pause-menu scene, and in-game scene of the game to operate the system and check if there was any human realizable delay between the request and the correct corresponding respond from the system.

   Initial State: The system is initially at the "Pre-Game State"

   Input: A development team member, A list of all possible and reasonable requests could be sent to the system

   Expected Output: There is not any presence of human realizable delay for any operation done to the system during the test.

   Result: PASS

2. NFS-P-2

Description: To test the availability of the system, one of the members of the development team manually launched the game with operating systems including Windows, Mac OS, and Linux then played the game for one time on from start playing the game to ”Death” of the space shooter on each of the operating systems in order to notice Space Shooter bugs and issues that were specific to any of the operating systems.

Initial State: The system is initially closed.

Input: A member of the development team; Windows, Mac OS, Linux operating system

Expected Output: There is not any presence of bug or issue that was specific to any of the operating systems told in the description during the test.

Result: PASS

2.4 Maintainability

1. NFS-M-1

   Description: The development team inspected the source code of the system to test if there is only one characteristic of the system that is likely to change encapsulated in each module.

   Initial State: The system is initially closed

   Input: Source code of the system, Development Team(Inspectors)

   Expected Output: Each module of the source code only contains one characteristic that is likely to change in the future.

   Result: PASS

2. NFS-M-2

   Description: The development team inspected the source code of the system to test if every module, class, method and global variable in the source code is documented properly with comment blocks.

   Initial State: The system is initially closed

Input: Source code of the system, Development Team(Inspectors)

Expected Output: The code is documented according to the standard so that documentation can be generated with the support of doxygen without any presence of error and texts that were used for documenting the code were descriptive, concise and correct.

Result: PASS

2.5 Cultural Requirement

1. NFS-C-1

   Description: The development team inspected the source code of the system to test if there is any asset(images, sounds) that is used in the implementation or displayed text on the general GUI of the system which would have any potential to be considered as offensive to any culture.

   Initial State: The system is initially closed

   Input: Source code, Sounds and images referenced in the source code of the system, Development Team(Inspectors)

   Expected Output: There is not asset that is used in the source code and text which could be displayed on the general GUI that could be considered as offensive to any culture.

   Result: PASS

3 Unit Testing

Unit Testing is a major part of the test suite for SpaceShooter and it is used because it excellently isolates and tests individual aspects of the program. This type of testing ensures that program runs without hiccups at the lowest level. Testing major aspects of the program should not be done before unit testing, as an error could be caused by many different methods in the program. Unit testing ensures that the program is completing its most basic functionalities perfectly.

4 Changes Due to Testing

After completing the Spaceship Movement Operation tests (FS-M-1 and FS-M-2) it was detected that the top and bottom borders had not been set after the implementation of the up and down movement capabilities. These tests exposed that the spaceship could move beyond the visible top and bottom borders. After this realization, the top and bottom physical borders were implemented and the ship now cannot go beyond what is visible.

The initial implementation of the new alien obstacles did not function correctly as the spaceship did not lose health after collision with is. This fault was exposed by conducting the Collision Detection test (FS-CD-2). After discovering this error, the Alien class was corrected.

All other parts of the program that were tested passed their respective tests.

5 Automated Testing

As previously mentioned in the Test Plan document, the test team has decided not to implement any automated testing mechanisms. This decision was made considering the lack of canvas environment testing framework support and the strict time constraints. The test team felt that the time would be better used for the implementation of the game and for manual testing.

6 Trace to Requirements

Test	Requirements
Functional Requirements Testing	
FS-MM-1	F2, F3, F4
FS-MM-2	F2, F3, F5
FS-MM-3	F2, F3, F4
FS-MM-4	F2, F3, F5
FS-MM-5	F2, F3, F5
FS-P-1	F2, F6
FS-P-2	F2, F7
FS-P-3	F2, F6
FS-P-4	F2, F6, F7
FS-P-5	F2, F6
FS-M-1	F13
FS-M-2	F13
FS-M-3	F13
FS-S-1	F14
FS-CD-1	F11, F12
FS-CD-2	F11, F12
FS-CD-3	F11, F12
FS-CD-4	F11, F12
FS-CD-5	F11, F12
FS-CD-6	F11, F12
FS-D-1	F15
FS-D-2	F15
Non-functional Requirements Testing	
NFS-LF-1	NF1, NF2
NFS-UH-1	NF3, NF4
NFS-P-1	NF5
NFS-P-2	NF7
NFS-M-1	NF12, NF13
NFS-M-2	NF12, NF15
NFS-C-1	NF17

Table 2: Trace Between Tests and Requirements

7 Trace to Modules

Test	Modules
Functional Requirements Testing	
FS-MM-1	M2, M3
FS-MM-2	M2, M3
FS-MM-3	M2, M3
FS-MM-4	M2, M3
FS-MM-5	M2, M3
FS-P-1	M2
FS-P-2	M2
FS-P-3	M2
FS-P-4	M2
FS-P-5	M2
FS-M-1	M4
FS-M-2	M4
FS-M-3	M4
FS-S-1	M4
FS-CD-1	M2, M4
FS-CD-2	M2, M4
FS-CD-3	M2, M4
FS-CD-4	M2, M4
FS-CD-5	M2, M4
FS-CD-6	M2, M4
FS-D-1	M2, M4
Non-functional Requirements Testing	
NFS-LF-1	M3, M6
NFS-UH-1	M2, M6
NFS-P-1	M1
NFS-P-2	M1
NFS-M-1	All Modules
NFS-M-2	All Modules
NFS-C-1	M5, M6

Table 3: Trace Between Tests and Modules

8 Code Coverage Metrics

The test suite designed by the test team produces roughly 93 percent code coverage through the tests. This is proven true given that all modules are covered through the tests, and this is confirmed above in the trace to modules section. The test suite tests all major components of the game, including all game functions and user interaction capabilities. The most important modules, which are the Game State and Game Objects modules, are thoroughly tested to ensure that the game runs without any errors or hiccups.