Test Plan

Gill, Surinder 1308896
Hu, Joshua 1311940
Lago, Nick 1302613

December 8, 2015

Contents

1 Revision History **14**

2 Introduction **15**
2.1 Purpose . . . 15
2.2 Environment and Pretest Background . . . 15
2.3 Test Objectives . . . 15
2.4 Expected Defect Rates . . . 15
2.5 References . . . 15

3 Plan **16**
3.1 Software Description . . . 16
3.2 Test Team . . . 16
3.3 Milestones . . . 17
3.4 Testing: Primary . . . 17
3.5 Testing: Secondary . . . 18

4 Structural Testing **19**
4.1 Appearance Testing . . . 19
4.1.1 Test Factors Involved . . . 19
4.1.2 Initial State . . . 19
4.1.3 Inputs . . . 19
4.1.4 Outputs . . . 19
4.1.5 Schedule . . . 19
4.1.6 Methodology . . . 19
4.1.7 Test For . . . 19
4.2 Style Testing . . . 19
4.2.1 Test Factors Involved . . . 19
4.2.2 Initial State . . . 20
4.2.3 Inputs . . . 20
4.2.4 Outputs . . . 20
4.2.5 Schedule . . . 20
4.2.6 Methodology . . . 20
4.2.7 Test For . . . 20
4.3 Ease of Use Testing . . . 20
4.3.1 Test Factors Involved . . . 20
4.3.2 Initial State . . . 20
4.3.3 Inputs . . . 20
4.3.4 Outputs . . . 20
4.3.5 Schedule . . . 21

4.3.6 Methodology . . . 21
4.3.7 Test For . . . 21
4.4 Learning Testing . . . 21
4.4.1 Test Factors Involved . . . 21
4.4.2 Initial State . . . 21
4.4.3 Inputs . . . 21
4.4.4 Outputs . . . 21
4.4.5 Schedule . . . 21
4.4.6 Methodology . . . 21
4.4.7 Test For . . . 21
4.5 Understandability Testing . . . 22
4.5.1 Test Factors Involved . . . 22
4.5.2 Initial State . . . 22
4.5.3 Inputs . . . 22
4.5.4 Outputs . . . 22
4.5.5 Schedule . . . 22
4.5.6 Methodology . . . 22
4.5.7 Test For . . . 22
4.6 Accessibility, and Interfacing with Adjacent Systems Testing . . . 22
4.6.1 Test Factors Involved . . . 22
4.6.2 Initial State . . . 22
4.6.3 Inputs . . . 23
4.6.4 Outputs . . . 23
4.6.5 Schedule . . . 23
4.6.6 Methodology . . . 23
4.6.7 Test For . . . 23
4.7 Speed and Latency Testing . . . 23
4.7.1 Test Factors Involved . . . 23
4.7.2 Initial State . . . 23
4.7.3 Inputs . . . 23
4.7.4 Outputs . . . 23
4.7.5 Schedule . . . 23
4.7.6 Methodology . . . 24
4.7.7 Test For . . . 24
4.8 Reliability and Availability, and Expected Physical Environment Testing . . . 24
4.8.1 Test Factors Involved . . . 24
4.8.2 Initial State . . . 24
4.8.3 Inputs . . . 24
4.8.4 Outputs . . . 24
4.8.5 Schedule . . . 24
4.8.6 Methodology . . . 24

4.8.7 Test For . . . 24
4.9 Robustness Testing . . . 25
4.9.1 Test Factors Involved . . . 25
4.9.2 Initial State . . . 25
4.9.3 Inputs . . . 25
4.9.4 Outputs . . . 25
4.9.5 Schedule . . . 25
4.9.6 Methodology . . . 25
4.9.7 Test For . . . 25
4.10 Capacity, and Instability Testing . . . 25
4.10.1 Test Factors Involved . . . 25
4.10.2 Initial State . . . 25
4.10.3 Inputs . . . 26
4.10.4 Outputs . . . 26
4.10.5 Schedule . . . 26
4.10.6 Methodology . . . 26
4.10.7 Test For . . . 26
4.11 Scalability Testing . . . 26
4.11.1 Test Factors Involved . . . 26
4.11.2 Initial State . . . 26
4.11.3 Inputs . . . 26
4.11.4 Outputs . . . 26
4.11.5 Schedule . . . 26
4.11.6 Methodology . . . 27
4.11.7 Test For . . . 27
4.12 Longevity Testing . . . 27
4.12.1 Test Factors Involved . . . 27
4.12.2 Initial State . . . 27
4.12.3 Inputs . . . 27
4.12.4 Outputs . . . 27
4.12.5 Schedule . . . 27
4.12.6 Methodology . . . 27
4.12.7 Test For . . . 27
4.13 Productization Testing . . . 27
4.13.1 Test Factors Involved . . . 28
4.13.2 Initial State . . . 28
4.13.3 Inputs . . . 28
4.13.4 Outputs . . . 28
4.13.5 Schedule . . . 28
4.13.6 Methodology . . . 28
4.13.7 Test For . . . 28

4.14 Supportability, Adaptability and Access Testing . . . 28
4.14.1 Test Factors Involved . . . 28
4.14.2 Initial State . . . 28
4.14.3 Inputs . . . 28
4.14.4 Outputs . . . 29
4.14.5 Schedule . . . 29
4.14.6 Methodology . . . 29
4.14.7 Test For . . . 29
4.15 Integrity Testing . . . 29
4.15.1 Test Factors Involved . . . 29
4.15.2 Initial State . . . 29
4.15.3 Inputs . . . 29
4.15.4 Outputs . . . 29
4.15.5 Schedule . . . 29
4.15.6 Methodology . . . 29
4.15.7 Test For . . . 30
4.16 Cultural, Compliance and Standards Testing . . . 30
4.16.1 Test Factors Involved . . . 30
4.16.2 Initial State . . . 30
4.16.3 Inputs . . . 30
4.16.4 Outputs . . . 30
4.16.5 Schedule . . . 30
4.16.6 Methodology . . . 30
4.16.7 Test For . . . 30

5 Specifications and Evaluation 31
5.1 Methods and Constraints . . . 32
5.2 Evaluation . . . 33

6 Test Descriptions 34
6.1 Test Identification . . . 34
6.2 Additional Test Identification . . . 34

7 Unit-Testing 35
7.1 Cookie Handler Setter Testing . . . 35
7.1.1 Test Factors Involved . . . 35
7.1.2 Initial State . . . 35
7.1.3 Inputs . . . 35
7.1.4 Outputs . . . 35
7.1.5 Schedule . . . 35
7.1.6 Methodology . . . 36

7.1.7 Tests For . . . 36
7.2 Cookie Handler Checker Testing . . . 36
7.2.1 Test Factors Involved . . . 36
7.2.2 Initial State . . . 36
7.2.3 Inputs . . . 36
7.2.4 Outputs . . . 36
7.2.5 Schedule . . . 36
7.2.6 Methodology . . . 36
7.2.7 Tests For . . . 36
7.3 Cookie Handler Getter Testing . . . 37
7.3.1 Test Factors Involved . . . 37
7.3.2 Initial State . . . 37
7.3.3 Inputs . . . 37
7.3.4 Outputs . . . 37
7.3.5 Schedule . . . 37
7.3.6 Methodology . . . 37
7.3.7 Tests For . . . 37
7.4 Cookie Handler Getter Testing . . . 37
7.4.1 Test Factors Involved . . . 37
7.4.2 Initial State . . . 38
7.4.3 Inputs . . . 38
7.4.4 Outputs . . . 38
7.4.5 Schedule . . . 38
7.4.6 Methodology . . . 38
7.4.7 Tests For . . . 38
7.5 Background Content Testing . . . 38
7.5.1 Test Factors Involved . . . 38
7.5.2 Initial State . . . 38
7.5.3 Inputs . . . 38
7.5.4 Outputs . . . 39
7.5.5 Schedule . . . 39
7.5.6 Methodology . . . 39
7.5.7 Tests For . . . 39
7.6 Background Coordinate Testing . . . 39
7.6.1 Test Factors Involved . . . 39
7.6.2 Initial State . . . 39
7.6.3 Inputs . . . 39
7.6.4 Outputs . . . 39
7.6.5 Schedule . . . 39
7.6.6 Methodology . . . 40
7.6.7 Tests For . . . 40

7.7 Collision Particle Testing . . . 40
7.7.1 Test Factors Involved . . . 40
7.7.2 Initial State . . . 40
7.7.3 Inputs . . . 40
7.7.4 Outputs . . . 40
7.7.5 Schedule . . . 40
7.7.6 Methodology . . . 40
7.7.7 Tests For . . . 40
7.8 Collision Matching Testing . . . 41
7.8.1 Test Factors Involved . . . 41
7.8.2 Initial State . . . 41
7.8.3 Inputs . . . 41
7.8.4 Outputs . . . 41
7.8.5 Schedule . . . 41
7.8.6 Methodology . . . 41
7.8.7 Tests For . . . 41
7.9 Collision Null Testing . . . 41
7.9.1 Test Factors Involved . . . 41
7.9.2 Initial State . . . 42
7.9.3 Inputs . . . 42
7.9.4 Outputs . . . 42
7.9.5 Schedule . . . 42
7.9.6 Methodology . . . 42
7.9.7 Tests For . . . 42
7.10 PowerUp Collision Particle Testing . . . 42
7.10.1 Test Factors Involved . . . 42
7.10.2 Initial State . . . 42
7.10.3 Inputs . . . 42
7.10.4 Outputs . . . 43
7.10.5 Schedule . . . 43
7.10.6 Methodology . . . 43
7.10.7 Tests For . . . 43
7.11 PowerUp Collision Matching Testing . . . 43
7.11.1 Test Factors Involved . . . 43
7.11.2 Initial State . . . 43
7.11.3 Inputs . . . 43
7.11.4 Outputs . . . 43
7.11.5 Schedule . . . 43
7.11.6 Methodology . . . 44
7.11.7 Tests For . . . 44
7.12 PowerUp Collision Null Testing . . . 44

7.12.1 Test Factors Involved . . . 44
7.12.2 Initial State . . . 44
7.12.3 Inputs . . . 44
7.12.4 Outputs . . . 44
7.12.5 Schedule . . . 44
7.12.6 Methodology . . . 44
7.12.7 Tests For . . . 44
7.13 Draw Rectangle Testing . . . 45
7.13.1 Test Factors Involved . . . 45
7.13.2 Initial State . . . 45
7.13.3 Inputs . . . 45
7.13.4 Outputs . . . 45
7.13.5 Schedule . . . 45
7.13.6 Methodology . . . 45
7.13.7 Tests For . . . 45
7.14 Draw Circle Testing . . . 45
7.14.1 Test Factors Involved . . . 45
7.14.2 Initial State . . . 46
7.14.3 Inputs . . . 46
7.14.4 Outputs . . . 46
7.14.5 Schedule . . . 46
7.14.6 Methodology . . . 46
7.14.7 Tests For . . . 46
7.15 Draw Image Testing . . . 46
7.15.1 Test Factors Involved . . . 46
7.15.2 Initial State . . . 46
7.15.3 Inputs . . . 46
7.15.4 Outputs . . . 47
7.15.5 Schedule . . . 47
7.15.6 Methodology . . . 47
7.15.7 Tests For . . . 47
7.16 Draw Sprite Testing . . . 47
7.16.1 Test Factors Involved . . . 47
7.16.2 Initial State . . . 47
7.16.3 Inputs . . . 47
7.16.4 Outputs . . . 47
7.16.5 Schedule . . . 47
7.16.6 Methodology . . . 48
7.16.7 Tests For . . . 48
7.17 Draw Text Testing . . . 48
7.17.1 Test Factors Involved . . . 48

7.17.2 Initial State . . . 48
7.17.3 Inputs . . . 48
7.17.4 Outputs . . . 48
7.17.5 Schedule . . . 48
7.17.6 Methodology . . . 48
7.17.7 Tests For . . . 48
7.18 Fish Image Function Testing . . . 49
7.18.1 Test Factors Involved . . . 49
7.18.2 Initial State . . . 49
7.18.3 Inputs . . . 49
7.18.4 Outputs . . . 49
7.18.5 Schedule . . . 49
7.18.6 Methodology . . . 49
7.18.7 Tests For . . . 49
7.19 Fish Gravity Function Testing . . . 49
7.19.1 Test Factors Involved . . . 49
7.19.2 Initial State . . . 50
7.19.3 Inputs . . . 50
7.19.4 Outputs . . . 50
7.19.5 Schedule . . . 50
7.19.6 Methodology . . . 50
7.19.7 Tests For . . . 50
7.20 Fish Velocity Function Testing . . . 50
7.20.1 Test Factors Involved . . . 50
7.20.2 Initial State . . . 50
7.20.3 Inputs . . . 50
7.20.4 Outputs . . . 51
7.20.5 Schedule . . . 51
7.20.6 Methodology . . . 51
7.20.7 Tests For . . . 51
7.21 User Input Tap Testing . . . 51
7.21.1 Test Factors Involved . . . 51
7.21.2 Initial State . . . 51
7.21.3 Inputs . . . 51
7.21.4 Outputs . . . 51
7.21.5 Schedule . . . 51
7.21.6 Methodology . . . 52
7.21.7 Tests For . . . 52
7.22 Jump Buffer Testing . . . 52
7.22.1 Test Factors Involved . . . 52
7.22.2 Initial State . . . 52

7.22.3 Inputs . . . 52
7.22.4 Outputs . . . 52
7.22.5 Schedule . . . 52
7.22.6 Methodology . . . 52
7.22.7 Tests For . . . 52
7.23 Medal Object Testing . . . 53
7.23.1 Test Factors Involved . . . 53
7.23.2 Initial State . . . 53
7.23.3 Inputs . . . 53
7.23.4 Outputs . . . 53
7.23.5 Schedule . . . 53
7.23.6 Methodology . . . 53
7.23.7 Tests For . . . 53
7.24 Correct Play Again Inputs Testing . . . 53
7.24.1 Test Factors Involved . . . 53
7.24.2 Initial State . . . 54
7.24.3 Inputs . . . 54
7.24.4 Outputs . . . 54
7.24.5 Schedule . . . 54
7.24.6 Methodology . . . 54
7.24.7 Tests For . . . 54
7.25 Cookie Access Testing . . . 54
7.25.1 Test Factors Involved . . . 54
7.25.2 Initial State . . . 54
7.25.3 Inputs . . . 54
7.25.4 Outputs . . . 55
7.25.5 Schedule . . . 55
7.25.6 Methodology . . . 55
7.25.7 Tests For . . . 55
7.26 Input Testing . . . 55
7.26.1 Test Factors Involved . . . 55
7.26.2 Initial State . . . 55
7.26.3 Inputs . . . 55
7.26.4 Outputs . . . 55
7.26.5 Schedule . . . 55
7.26.6 Methodology . . . 56
7.26.7 Tests For . . . 56
7.27 Input Null Testing . . . 56
7.27.1 Test Factors Involved . . . 56
7.27.2 Initial State . . . 56
7.27.3 Inputs . . . 56

7.27.4 Outputs . . . 56
7.27.5 Schedule . . . 56
7.27.6 Methodology . . . 56
7.27.7 Tests For . . . 56
7.28 Coin Testing . . . 57
7.28.1 Test Factors Involved . . . 57
7.28.2 Initial State . . . 57
7.28.3 Inputs . . . 57
7.28.4 Outputs . . . 57
7.28.5 Schedule . . . 57
7.28.6 Methodology . . . 57
7.28.7 Tests For . . . 57
7.29 Image Rendering Testing . . . 57
7.29.1 Test Factors Involved . . . 57
7.29.2 Initial State . . . 58
7.29.3 Inputs . . . 58
7.29.4 Outputs . . . 58
7.29.5 Schedule . . . 58
7.29.6 Methodology . . . 58
7.29.7 Tests For . . . 58
7.30 Random Integer Range Testing . . . 58
7.30.1 Test Factors Involved . . . 58
7.30.2 Initial State . . . 58
7.30.3 Inputs . . . 58
7.30.4 Outputs . . . 59
7.30.5 Schedule . . . 59
7.30.6 Methodology . . . 59
7.30.7 Tests For . . . 59
7.31 Splash Rendering Testing . . . 59
7.31.1 Test Factors Involved . . . 59
7.31.2 Initial State . . . 59
7.31.3 Inputs . . . 59
7.31.4 Outputs . . . 59
7.31.5 Schedule . . . 59
7.31.6 Methodology . . . 60
7.31.7 Tests For . . . 60
7.32 Audio Sample Testing . . . 60
7.32.1 Test Factors Involved . . . 60
7.32.2 Initial State . . . 60
7.32.3 Inputs . . . 60
7.32.4 Outputs . . . 60

7.32.5 Schedule . . . 60
7.32.6 Methodology . . . 60
7.32.7 Tests For . . . 60
7.33 Entities Testing . . . 61
7.33.1 Test Factors Involved . . . 61
7.33.2 Initial State . . . 61
7.33.3 Inputs . . . 61
7.33.4 Outputs . . . 61
7.33.5 Schedule . . . 61
7.33.6 Methodology . . . 61
7.33.7 Tests For . . . 61
7.34 Canvas Testing . . . 61
7.34.1 Test Factors Involved . . . 61
7.34.2 Initial State . . . 62
7.34.3 Inputs . . . 62
7.34.4 Outputs . . . 62
7.34.5 Schedule . . . 62
7.34.6 Methodology . . . 62
7.34.7 Tests For . . . 62
7.35 Canvas Testing . . . 62
7.35.1 Test Factors Involved . . . 62
7.35.2 Initial State . . . 62
7.35.3 Inputs . . . 62
7.35.4 Outputs . . . 63
7.35.5 Schedule . . . 63
7.35.6 Methodology . . . 63
7.35.7 Tests For . . . 63
7.36 Update Error Testing . . . 63
7.36.1 Test Factors Involved . . . 63
7.36.2 Initial State . . . 63
7.36.3 Inputs . . . 63
7.36.4 Outputs . . . 63
7.36.5 Schedule . . . 63
7.36.6 Methodology . . . 64
7.36.7 Tests For . . . 64
7.37 Sound Play Testing . . . 64
7.37.1 Test Factors Involved . . . 64
7.37.2 Initial State . . . 64
7.37.3 Inputs . . . 64
7.37.4 Outputs . . . 64
7.37.5 Schedule . . . 64

7.37.6 Methodology . . . 64
7.37.7 Tests For . . . 64
7.38 Sound Time Testing . . . 65
7.38.1 Test Factors Involved . . . 65
7.38.2 Initial State . . . 65
7.38.3 Inputs . . . 65
7.38.4 Outputs . . . 65
7.38.5 Schedule . . . 65
7.38.6 Methodology . . . 65
7.38.7 Tests For . . . 65
7.39 Animation Frame Testing . . . 65
7.39.1 Test Factors Involved . . . 65
7.39.2 Initial State . . . 66
7.39.3 Inputs . . . 66
7.39.4 Outputs . . . 66
7.39.5 Schedule . . . 66
7.39.6 Methodology . . . 66
7.39.7 Tests For . . . 66

List of Figures

1 Software Description . . . 16

List of Tables

1 Revision History: Test Plan . . . 14
2 Milestones . . . 17

1 Revision History

Table 1: Revision History: Test Plan

October 21, 2015	**DEVELOPER**	**CHANGE**	**REVISION**
October 21, 2015	Gill, Surinder	Sections: 2	0
October 21, 2015	Hu, Joshua	Sections: 3	0
October 21, 2015	Lago, Nick	Sections: 1, 4	0
December 7, 2015	Gill, Surinder	Sections: 1-5	1
December 7, 2015	Hu, Joshua	Sections: 1-5	1
December 7, 2015	Lago, Nick	Sections: 1-5	1

2 Introduction

2.1 Purpose

The function of our program is to run a 2D survival arcade style game. This game will have to play sounds, render graphics and interpret user inputs. The testing for this software will include the Jasmine Framework as our automated unit testing. For structural testing our group will implement the Jasmine Framework to test collision detection, user input recognition, and cookie storing. Functional testing will consist of sound output testing along with sprite rendering and frame resizing. Mutation testing will also occur on our collision detection function as we alter the parameters for optimal sensitivity and to check to make sure that no other function is altered in the process. Fault testing will occur throughout the design and implementation of this project.

2.2 Environment and Pretest Background

Since this is a brand new project there has been no previous testing. Since this is similar to a popular game called Flappy Bird we do have expected outputs for all the testing that will occur.

2.3 Test Objectives

Testing will allow our group to make sure that the full functionality of our game is being met. Before releasing our software all of our tests must be thoroughly completed and passed. This grants our team the peace of mind that all functionality of our program is working properly.

2.4 Expected Defect Rates

We estimate that no defects will take place during the testing and no further testing will be needed after this set.

2.5 References

There is no previously published documentation on this project. There is documentation however on related documents called "Flappy Bird" which should be more than enough: `http://flappybird.wikia.com/wiki/Category:Template_documentation`

3 Plan

3.1 Software Description

Inputs	Outputs	Functions of Software being tested
Keyboard Input – The game will be relying on keyboard inputs for the two player mode that will be implemented	Server host – The game code will be outputted to a server which will host our website and can be accessed through the world wide web from a display	Cookie Function – This function will hold memory of recent high scores of previous players. It will allow the user to close the game from the browser and access previous scores when reopened if it allows cookies.
Mouse Input – The game will be relying on mouse input for the one player game mode		Sound Function – This function will be importing all the sounds required for the game and then set channels for them to play.
		Window Setter (output) Function - This function looks for different types of devices and has it return the type of device that it should output for.
		Input Function – This function recognizes different inputs by the user and returns the type of input.
		Pipes Function – This function creates the pipes and allows them to interact with other characters and objects in the setting.
		Sprites Function – This function creates different sprites ranging from underwater objects to the main fish object.
		Collision Function – This function detects collisions made by the main sprite and the pipes created
		Menu Function – This function distinguishes between different menus and allows the user to pick one player or two player
		Main play Function – This function creates the game and connects the different function together. This function will be used when using black box/functional testing.

Figure 1: Software Description

3.2 Test Team

The test team will consist of Surinder Gill, Nick Lago and Joshua Hu. The three main testers will split the entire testing evenly covering all the different types of testing and

preparing the automated testing.

3.3 Milestones

The locations for testing will be centralized in our designated lab locations and work areas. Milestones and dates for the testing will be based off the different tests that will be conducted:

Table 2: Milestones

Module Testing	**Expected Date of Completion**
Sound Output testing	October 30, 2015
Frame resizing testing	October 30, 2015
Input data testing	November 6, 2015
Screen Output (Rendering) testing	November 6, 2015
Sequencing order testing	November 13, 2015
Sprite Rendering testing	November 13, 2015
Cookie storing testing	November 20, 2015
Restart and loop testing	November 20, 2015

3.4 Testing: Primary

The software that will be required is a JavaScript automated testing framework, a server that can host our website when we're testing on different devices and browsers, a text editor that can make changes to specific pieces of code that will require changing, and a media editor that will be able to modify images and change signs. The required personnel to set-up automated testing will be required to have sufficient knowledge in JavaScript and HTML to setup multiple test cases. However, users and development groups that will be doing manual testing do not not have specific requirements needed for testing. The automated testing will be done through Jasmine Framework that provides its own documentation. It creates cases that require inputs and expected outputs and will allow the tester to find faults if any. It will create webpages with documentation that show whether a test case has passed or failed and why. The proof of concept will indicate correct sound output and frame resizing. It will display a simple test run through the Jasmine Framework showing that both functions are ideal in their outputs. This testing will then be shown and used as a guideline to other tests.

3.5 Testing: Secondary

Secondary testing will be done by users in our open beta testing. We will allow users to access our game at an early development stage after preliminary tests by developers. They will be able to manually perform functional testing by checking for bugs and running through the game through our website.

4 Structural Testing

4.1 Appearance Testing

This will be a structural test by running the game.

4.1.1 Test Factors Involved

Correctness

4.1.2 Initial State

Application is running

4.1.3 Inputs

User inputs

4.1.4 Outputs

Renders logo, underwater scene background and other entities to catch users eye.

4.1.5 Schedule

This feature will be implemented for Final Demonstration 1

4.1.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey

4.1.7 Test For

This test validates section 5.1.1, of requirements documentation.

4.2 Style Testing

This will be a structural test by running the game.

4.2.1 Test Factors Involved

Correctness

4.2.2 Initial State

Application is running

4.2.3 Inputs

User inputs

4.2.4 Outputs

Rendering entities to screen

4.2.5 Schedule

This feature will be implemented for Final Demonstration 1

4.2.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.2.7 Test For

This test validates section 5.1.2, of requirements documentation.

4.3 Ease of Use Testing

This will be a structural test by running the game.

4.3.1 Test Factors Involved

Ease of Use

4.3.2 Initial State

Application is running

4.3.3 Inputs

User inputs

4.3.4 Outputs

Playability factors based off of users feedback

4.3.5 Schedule

This feature will be implemented for Final Demonstration 1

4.3.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.3.7 Test For

This test validates section 5.1.2, of requirements documentation.

4.4 Learning Testing

This will be a structural test by running the game.

4.4.1 Test Factors Involved

Ease of Use

4.4.2 Initial State

Application is running

4.4.3 Inputs

User inputs

4.4.4 Outputs

Ease of use factors made users playing with a mouse

4.4.5 Schedule

This feature will be implemented for Final Demonstration 1

4.4.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.4.7 Test For

This test validates section 5.2.3, of requirements documentation.

4.5 Understandability Testing

This will be a structural test by running the game.

4.5.1 Test Factors Involved

Ease of Use

4.5.2 Initial State

Application is running

4.5.3 Inputs

User inputs

4.5.4 Outputs

Understandability factors based off of users feedback

4.5.5 Schedule

This feature will be implemented for Final Demonstration 1

4.5.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.5.7 Test For

This test validates section 5.2.4, of requirements documentation.

4.6 Accessibility, and Interfacing with Adjacent Systems Testing

This will be a structural test by running the game.

4.6.1 Test Factors Involved

Ease of Use

4.6.2 Initial State

Application is activated

4.6.3 Inputs

Run application on different browsers and devices

4.6.4 Outputs

Application initialization and playability determined by users

4.6.5 Schedule

This feature will be implemented for Final Demonstration 1

4.6.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.6.7 Test For

This test validates section 5.2.5, and 5.4.2 of requirements documentation.

4.7 Speed and Latency Testing

This will be a structural test by running the game.

4.7.1 Test Factors Involved

Ease of Use

4.7.2 Initial State

Application is activated

4.7.3 Inputs

User inputs

4.7.4 Outputs

Application response to user inputs

4.7.5 Schedule

This feature will be implemented for Final Demonstration 1

4.7.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.7.7 Test For

This test validates section 5.3.1, of requirements documentation.

4.8 Reliability and Availability, and Expected Physical Environment Testing

This will be a structural test by running the game.

4.8.1 Test Factors Involved

Ease of Use

4.8.2 Initial State

Application is activated

4.8.3 Inputs

User to activate game at various locations where access to internet is offered.

4.8.4 Outputs

Application runs

4.8.5 Schedule

This feature will be implemented for Final Demonstration 1

4.8.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.8.7 Test For

This test validates section 5.3.4 and 5.4.1, of requirements documentation.

4.9 Robustness Testing

This will be a structural test by running the game.

4.9.1 Test Factors Involved

Correctness

4.9.2 Initial State

Application is run

4.9.3 Inputs

User presses input multiple times in a row.

4.9.4 Outputs

Application runs smoothly

4.9.5 Schedule

This feature will be implemented for Final Demonstration 1

4.9.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.9.7 Test For

This test validates section 5.3.5, of requirements documentation.

4.10 Capacity, and Instability Testing

This will be a structural test by running the application.

4.10.1 Test Factors Involved

Ease of Use

4.10.2 Initial State

Application is activated

4.10.3 Inputs

Multiple users will activate application.

4.10.4 Outputs

Application runs.

4.10.5 Schedule

This feature will be implemented for Final Demonstration 1

4.10.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.10.7 Test For

This test validates section 5.3.6, and 5.3.7of requirements documentation.

4.11 Scalability Testing

This will be a structural test by running the application.

4.11.1 Test Factors Involved

Correctness

4.11.2 Initial State

Application is inactive

4.11.3 Inputs

Developer will change state variables

4.11.4 Outputs

Application runs appropriately based on changes to code.

4.11.5 Schedule

This feature will be implemented for Final Demonstration 1

4.11.6 Methodology

Tests will be done manually by developers changing code and running the prototype.

4.11.7 Test For

This test validates section 5.3.8, of requirements documentation.

4.12 Longevity Testing

This will be a structural test by running the application.

4.12.1 Test Factors Involved

Correctness

4.12.2 Initial State

Application is activated

4.12.3 Inputs

User will play game multiple times in a row without closing browser.

4.12.4 Outputs

Game runs as expected for entire game time.

4.12.5 Schedule

This feature will be implemented for Final Demonstration 1

4.12.6 Methodology

Tests will be done manually by developers running prototype. Tests will be enhanced by actual user feedback via a survey.

4.12.7 Test For

This test validates section 5.3.9, of requirements documentation.

4.13 Productization Testing

This will be a structural test by running the application.

4.13.1 Test Factors Involved

Correctness

4.13.2 Initial State

Application is activated on developer Surinder's site

4.13.3 Inputs

Developer will launch game

4.13.4 Outputs

Application runs as developer deems it should.

4.13.5 Schedule

This feature will be implemented for Final Demonstration 1

4.13.6 Methodology

Tests will be done manually by developers running prototype.

4.13.7 Test For

This test validates section 5.4.3, of requirements documentation.

4.14 Supportability, Adaptability and Access Testing

This will be a structural test by running the application.

4.14.1 Test Factors Involved

Correctness

4.14.2 Initial State

Application is run

4.14.3 Inputs

Multiple consumers will launch game

4.14.4 Outputs

Application runs as consumer deems it should.

4.14.5 Schedule

This feature will be implemented for Final Demonstration 1

4.14.6 Methodology

Tests will be done manually by consumer running prototype and giving feedback via a survey.

4.14.7 Test For

This test validates section 5.5.2, 5.5.3 and 5.6.1 of requirements documentation.

4.15 Integrity Testing

This will be a structural test by running the application.

4.15.1 Test Factors Involved

Correctness

4.15.2 Initial State

Application is run

4.15.3 Inputs

Developer picks garbage inputs

4.15.4 Outputs

Application does not accept such input.

4.15.5 Schedule

This feature will be implemented for Final Demonstration 1

4.15.6 Methodology

Tests will be done manually by consumer running prototype.

4.15.7 Test For

This test validates section 5.of requirements documentation.

4.16 Cultural, Compliance and Standards Testing

This will be a structural test by running the application.

4.16.1 Test Factors Involved

Correctness

4.16.2 Initial State

Application is run

4.16.3 Inputs

Developer runs application

4.16.4 Outputs

No laws, licenses are broken. No cultures are offended.

4.16.5 Schedule

This feature will be implemented for Final Demonstration 1

4.16.6 Methodology

Tests will be done manually by consumer running prototype.

4.16.7 Test For

This test validates section 5.of requirements documentation.

5 Specifications and Evaluation

Business Functions

- The executable HTML file will create a new browser window.
- The HTML will be executed by a browser with JavaScript functionality.
- The game will have a standby state in which it waits for user input.
- Upon the reception of user input from the standby state the game will begin.
- At the beginning of the game the user will perceive all stats reset to their default state.
- At the beginning of the game the user character will maintain its state until user input is received.
- If there is a collision with the user character and an obstacle object the game will terminate and all stats will be recorded.
- Upon termination of the game state all stats will be reset to their default state and the standby state will be reinitiated.
- If there is a collision with the user character and an objective object the user's score will increment and the objective object's instance will terminate.
- During the game state reception of user input will cause the user character to respond in a constant and uniform manner relative to the user character's instance.

Structural Functions

- Cookie

  Test how cookies are created and used by the program through static and dynamic testing. This includes code analysis, unit testing, and system testing.

- Rendering

  Test the use of graphic files compared to drawing objects for the game rendering. This includes manual system testing for aesthetic purposes.

Test/Function Relationships

Much of the testing done for the structural functions will be changed depending on how we (the developers) choose to optimize or design the appearance of the product.

For the structural testing of the cookies, it must be decided on which data to store for each cookies instance and how that data will affect the game. This can be done initially through manual code analysis then verified with unit testing, and a manual system test where the value of the cookie can be checked by access through the game and of where the cookies are stored.

The structural rendering tests will be used to decide through the aesthetic decisions of the designers and developers the presentation of core functional and ornamental objects in the game. As such the tests conducted will mostly be done through manual system tests although additional unit tests can be performed to validate the functionality of either drawing or using images to render objects.

Test Progression

The tests will proceed by verifying the business functions and all critical components of the project software. With the basis of the project verified structural testing can proceed to optimize the performance of the product and the end users' experience with the product.

5.1 Methods and Constraints

Methodology

The A Team's (the developers) approach to testing is to validate the core functionality of the product being developed before testing non-critical components of the product.

Test Tools

For unit testing we will be using the testing framework Jasmine. For system testing we will be running the game by executing it on a browser and observing the functionalities. For code analysis we will use humans with coding experience to analyze the code.

Extent

The entirety of the product will be tested.

Data Recording

The results of the unit testing will be written to a test results log. System testing will be added to a separate test log specifcaly for system testing, as will code analysis.

All test will contain information on the aspect being tested, the date of the test, the person running the test, the results of the test, a description of the test purpose, a description of the test results, and next steps from the test.

Constraints

Due to the game's simple mechanics, there are few constraints on the testing.

5.2 Evaluation

Criteria

Our tests will cover primarily the boundaries of the game mechanics and some testing inside the boundaries as an example test of normal behaviour.

Data Reduction

All test logs will indicate a pass or a need fro review which allow us to focus on reimplementing and testing components which do not pass the tests.

6 Test Descriptions

6.1 Test Identification

Control: The team are going to be using automatic insertions, in the form of a unit tester named Jasmine. Our team will also manually create non automated inputs in the form of playing the game.

~~The inputs created from Jasmine will be in the three different forms. First will we have Jasmine change the location of our fish player to random locations, and by knowing what spaces are occupied by pipes, the Jasmine Framework can determine when the collision function should return true. Next our inputs will be in the form of Jasmine checking for stored cookies. As a result of some browsers not allowing cookies, this isn't a full proof test and some manual testing will be needed. Lastly, the input going into the input recognition function will be a replica of random inputs a user may put if they were actually playing. If the program registers the input as intended, the program will pass. For the manual testing, our inputs will be the team actually running the game and giving usual and unusual user inputs and seeing how the program reacts.~~

Manual testing will occur on different browsers such as Chrome, Explorer, Firefox, etc. This will allow diversity within our tests. Automated testing (through Jasmine) will consist of many different sets of inputs. Jasmine will be given a set of expected outputs and will allow a test to pass if these outputs are met given many different inputs. Some tests will attempt to generate exceptions by accessing inputs outside of the alphabet of possible inputs such as accessing cookies that do not exist, etc.

~~The outputs will be pretty straight forward to tell for the Jasmine Framework. If the fish is on the same location that is occupied by a pipe the collision function should return true. The output for the cookie testing will be slightly different, as it will just be a check to see if the cookies were stored. The output for our manual testing will be comprised of images showing the expected results~~.

6.2 Additional Test Identification

The last of our testing will be semantics and syntax testing. We need for this program to render neatly and look clean. This will be done by trying different frame sizes, different animations and changing the images in our sprite library. Checking this fairly regularly is a smart idea to find the most ascetically pleasing set of sprites and the nicest size to fit our game onto (a preferred size). Variable naming and implementation layouts will be tested to make sure our code is understandable to all members and that the code makes sense to someone trying to maintain it. Syntax will be tested regularly by running the program and having other members look over new additions. This will make sure that we avoid a

syntax mistake that could cause problems later down the road for our project and have the team unsure about its location. System tests will be conducted to cross check against the prototype to ensure proper functionality.

In addition structural testing will include performance, difficulty, accessibility, and meeting aesthetic requirements. Performance related testing will determine frame rate speeds on differing browsers; server lag and the machine's processing power will be taken into consideration. Difficulty will be adjusted for a subjectively optimal user experience by mutation testing the rendered obstacle functions. The accessibility will be tested by running system tests on different machine types running varying browsers and operating systems per test. Aesthetic requirements will be met through manual system testing by using different sprite designs and will be decided at the discretion of the developers.

7 Unit-Testing

7.1 Cookie Handler Setter Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.1.1 Test Factors Involved

Correctness

7.1.2 Initial State

No cookies stored at the moment

7.1.3 Inputs

Entering in the website address hosting the application. A cookie will be inserted that has a name, value and expiry date.

7.1.4 Outputs

A cookie will be stored on the user's browser that contains.

7.1.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.1.6 Methodology

This test will be conducted by setting a new cookie using the setCookie method in the cookies class. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.1.7 Tests For

This test validates section 4.4.1, of requirements documentation.

7.2 Cookie Handler Checker Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.2.1 Test Factors Involved

Correctness

7.2.2 Initial State

No cookies stored at the moment

7.2.3 Inputs

Entering in the website address hosting the application. A cookie will be inserted that has a name, value of 2 and expiry date.

7.2.4 Outputs

A cookie will be checked to see if there is a value of 2.

7.2.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.2.6 Methodology

This test will be conducted by setting a new cookie using the setCookie method and equals in the cookies class. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.2.7 Tests For

This test validates section 4.4.1, of requirements documentation.

7.3 Cookie Handler Getter Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.3.1 Test Factors Involved

Correctness

7.3.2 Initial State

Cookie stored with a value of 42.

7.3.3 Inputs

Entering in the website address hosting the application.

7.3.4 Outputs

A cookie will be checked that has a name, value of 42 and expiry date.

7.3.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.3.6 Methodology

This test will be conducted by setting a new cookie using the getCookie method and equals in the cookies class. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.3.7 Tests For

This test validates section 4.4.2, of requirements documentation.

7.4 Cookie Handler Getter Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.4.1 Test Factors Involved

Correctness

7.4.2 Initial State

Cookie stored with a value of 42.

7.4.3 Inputs

Entering in the website address hosting the application.

7.4.4 Outputs

A cookie will be checked that has a name, null value and expiry date.

7.4.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.4.6 Methodology

This test will be conducted by setting a new cookie using the getCookie method and equals in the cookies class. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.4.7 Tests For

This test validates section 4.4.2, of requirements documentation.

7.5 Background Content Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.5.1 Test Factors Involved

Correctness

7.5.2 Initial State

this.bg is initialized

7.5.3 Inputs

Input this.bg.src = bg.png from files

7.5.4 Outputs

Check to see if bg has an image attached to it.

7.5.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.5.6 Methodology

This test will be conducted by setting a picture to the background and check to see if it has done that. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.5.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.6 Background Coordinate Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.6.1 Test Factors Involved

Correctness

7.6.2 Initial State

this.x and this.y is initialized

7.6.3 Inputs

Inputs: x-coordinate, y-coordinate and radius

7.6.4 Outputs

Check to see if inputs have the correct x and y coordinates.

7.6.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.6.6 Methodology

This test will be conducted by checking for the inputs from the bottomBar function and checking to see if they are correct. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.6.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.7 Collision Particle Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.7.1 Test Factors Involved

Correctness

7.7.2 Initial State

Coordinates for fish have been initialized

7.7.3 Inputs

Inputs: fish and pipe objects

7.7.4 Outputs

Check to see if x and y coordinates of the fish are in contact with the x and y coordinates of the pipe.

7.7.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.7.6 Methodology

This test will be conducted by checking for the inputs from the collides function and checking to see if they interact. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.7.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.8 Collision Matching Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.8.1 Test Factors Involved

Correctness

7.8.2 Initial State

Coordinates for fish and pipe have been initialized

7.8.3 Inputs

Inputs: fish and pipe objects

7.8.4 Outputs

Check to see if x and y coordinates of the fish are in distance with the x and y coordinates of the pipe.

7.8.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.8.6 Methodology

This test will be conducted by checking for the inputs from the collides function and checking to see if they are in contact. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.8.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.9 Collision Null Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.9.1 Test Factors Involved

Correctness

7.9.2 Initial State

Coordinates for fish and pipe have been initialized

7.9.3 Inputs

Inputs: fish and pipe objects

7.9.4 Outputs

Check to see if the collision is null because there is no collision occurring.

7.9.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.9.6 Methodology

This test will be conducted by checking for the inputs from the collides function and confirming that a null is correct. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.9.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.10 PowerUp Collision Particle Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.10.1 Test Factors Involved

Correctness

7.10.2 Initial State

Coordinates for fish and powerup have been initialized

7.10.3 Inputs

Inputs: fish and powerup objects

7.10.4 Outputs

Check to see if x and y coordinates of the fish are in contact with the x and y coordinates of the powerup.

7.10.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.10.6 Methodology

This test will be conducted by checking for the inputs from the powerup collides function and checking to see if they interact. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.10.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.11 PowerUp Collision Matching Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.11.1 Test Factors Involved

Correctness

7.11.2 Initial State

Coordinates for fish and powerup have been initialized

7.11.3 Inputs

Inputs: fish and powerup objects

7.11.4 Outputs

Check to see if x and y coordinates of the fish are in distance with the x and y coordinates of the powerup.

7.11.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.11.6 Methodology

This test will be conducted by checking for the inputs from the powerup collides function and checking to see if they are in contact. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.11.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.12 PowerUp Collision Null Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.12.1 Test Factors Involved

Correctness

7.12.2 Initial State

Coordinates for fish and pipe have been initialized

7.12.3 Inputs

Inputs: fish and pipe objects

7.12.4 Outputs

Check to see if the collision is null because there is no collision occurring.

7.12.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.12.6 Methodology

This test will be conducted by checking for the inputs from the collides function and confirming that a null is correct. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.12.7 Tests For

This test validates section 4.4.9, of requirements documentation.

7.13 Draw Rectangle Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.13.1 Test Factors Involved

Correctness

7.13.2 Initial State

N/A

7.13.3 Inputs

Inputs: x, y, width, height, colour

7.13.4 Outputs

Check to see if the object is filled with the colour and appropriate dimensions.

7.13.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.13.6 Methodology

This test will be conducted by comparing the inputs from the draw rect function and confirming that all the variables have recieved the inputs. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.13.7 Tests For

This test validates section 4.4.14, of requirements documentation.

7.14 Draw Circle Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.14.1 Test Factors Involved

Correctness

7.14.2 Initial State

N/A

7.14.3 Inputs

Inputs: x, y, radius, colour

7.14.4 Outputs

Check to see if the object is filled with the colour and appropriate dimensions.

7.14.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.14.6 Methodology

This test will be conducted by comparing the inputs from the draw circle function and confirming that all the variables have received the inputs. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.14.7 Tests For

This test validates section 4.4.14, of requirements documentation.

7.15 Draw Image Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.15.1 Test Factors Involved

Correctness

7.15.2 Initial State

N/A

7.15.3 Inputs

Inputs: x, y, img

7.15.4 Outputs

Check to see if the object has loaded the correct image and appropriate dimensions.

7.15.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.15.6 Methodology

This test will be conducted by comparing the inputs from the draw image function and confirming that all the variables have recieved the inputs. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.15.7 Tests For

This test validates section 4.4.14, of requirements documentation.

7.16 Draw Sprite Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.16.1 Test Factors Involved

Correctness

7.16.2 Initial State

N/A

7.16.3 Inputs

Inputs: img, srcX, srcY, srcW, srcH, destX, destY, destW, destH, r

7.16.4 Outputs

Check to see if the object has loaded the correct image and appropriate dimensions.

7.16.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.16.6 Methodology

This test will be conducted by comparing the inputs from the draw sprite function and confirming that all the variables have recieved the inputs. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.16.7 Tests For

This test validates section 4.4.14, of requirements documentation.

7.17 Draw Text Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.17.1 Test Factors Involved

Correctness

7.17.2 Initial State

N/A

7.17.3 Inputs

Inputs: string, x, y, size, colour

7.17.4 Outputs

Check to see if the object has loaded the correct image, colour and appropriate dimensions.

7.17.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.17.6 Methodology

This test will be conducted by comparing the inputs from the draw text function and confirming that all the variables have recieved the inputs. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.17.7 Tests For

This test validates section 4.4.14, of requirements documentation.

7.18 Fish Image Function Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.18.1 Test Factors Involved

Correctness

7.18.2 Initial State

N/A

7.18.3 Inputs

There are no inputs

7.18.4 Outputs

Check to see if the image is initialized and loaded.

7.18.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.18.6 Methodology

This test will be conducted by initializing the bird picture and having it retrieve the png file from the images folder. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.18.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.19 Fish Gravity Function Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.19.1 Test Factors Involved

Correctness

7.19.2 Initial State

N/A

7.19.3 Inputs

There are no inputs

7.19.4 Outputs

Check to see if gravity is initialized and set to default value.

7.19.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.19.6 Methodology

This test will be conducted by initializing gravity and giving it a default value set by the developer. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.19.7 Tests For

This test validates section 4.4.16, of requirements documentation.

7.20 Fish Velocity Function Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.20.1 Test Factors Involved

Correctness

7.20.2 Initial State

N/A

7.20.3 Inputs

There are no inputs

7.20.4 Outputs

Check to see if velocity is initialized and set to default value.

7.20.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.20.6 Methodology

This test will be conducted by initializing velocity and giving it a default value set by the developer. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.20.7 Tests For

This test validates section 4.4.16, of requirements documentation.

7.21 User Input Tap Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.21.1 Test Factors Involved

Correctness

7.21.2 Initial State

N/A

7.21.3 Inputs

Input: User click or tap

7.21.4 Outputs

Check to see if user has clicked on the window.

7.21.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.21.6 Methodology

This test will be conducted by checking to see if the user click is not null. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.21.7 Tests For

This test validates section 4.4.12, of requirements documentation.

7.22 Jump Buffer Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.22.1 Test Factors Involved

Correctness

7.22.2 Initial State

N/A

7.22.3 Inputs

There are no inputs.

7.22.4 Outputs

Check that the jump buffer is set to -3.

7.22.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.22.6 Methodology

This test will be conducted by checking to see if the jump buffer is set to -3 but the equals function in Jasmine. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.22.7 Tests For

This test validates section 4.4.16, of requirements documentation.

7.23 Medal Object Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.23.1 Test Factors Involved

Correctness

7.23.2 Initial State

N/A

7.23.3 Inputs

The user's score.

7.23.4 Outputs

The medal colour based on score.

7.23.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.23.6 Methodology

This test will be conducted by checking to see if the correct medal is returned from GameOver function. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.23.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.24 Correct Play Again Inputs Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.24.1 Test Factors Involved

Correctness

7.24.2 Initial State

GameOver Mode

7.24.3 Inputs

Input: x and y coordinates

7.24.4 Outputs

Update to restart game.

7.24.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.24.6 Methodology

This test will be conducted by checking to see if the user has tapped within a certain area to restart the game. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.24.7 Tests For

This test validates section 4.4.10 and 4.4.11, of requirements documentation.

7.25 Cookie Access Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.25.1 Test Factors Involved

Correctness

7.25.2 Initial State

GameOver Mode

7.25.3 Inputs

Input: User's Highscore

7.25.4 Outputs

New Highscore when old highscore is lower

7.25.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.25.6 Methodology

This test will be conducted by making the highscore bigger and checking to see if it has changed. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.25.7 Tests For

This test validates section 4.4.2, of requirements documentation.

7.26 Input Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.26.1 Test Factors Involved

Correctness

7.26.2 Initial State

Game Running State

7.26.3 Inputs

Input: User's mouse click

7.26.4 Outputs

X and y coordinates from the user's mouse click.

7.26.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.26.6 Methodology

This test will be conducted by confirming that the user's click is recorded in x and y values. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.26.7 Tests For

This test validates section 4.4.12, of requirements documentation.

7.27 Input Null Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.27.1 Test Factors Involved

Correctness

7.27.2 Initial State

Game Running State

7.27.3 Inputs

Input: User's mouse click

7.27.4 Outputs

Null checker for user's mouse click.

7.27.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.27.6 Methodology

This test will be conducted by confirming that the user's click is not null during the click. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.27.7 Tests For

This test validates section 4.4.12, of requirements documentation.

7.28 Coin Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.28.1 Test Factors Involved

Correctness

7.28.2 Initial State

Game Running State

7.28.3 Inputs

There is no input

7.28.4 Outputs

Location of coin in retrospect to pipe

7.28.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.28.6 Methodology

This test will be conducted by assessing the location of the coin in the frame based off coordinates. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.28.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.29 Image Rendering Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.29.1 Test Factors Involved

Correctness

7.29.2 Initial State

Game Running State

7.29.3 Inputs

Input: Source image from root folders.

7.29.4 Outputs

Variables are assigned to particular images and have been initialized.

7.29.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.29.6 Methodology

This test will be conducted by assessing the location of the coin in the frame based off coordinates. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.29.7 Tests For

This test validates section 4.4.13, of requirements documentation.

7.30 Random Integer Range Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.30.1 Test Factors Involved

Correctness

7.30.2 Initial State

Game Running State

7.30.3 Inputs

Input: random integer within domain

7.30.4 Outputs

Integer within a certain domain.

7.30.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.30.6 Methodology

This test will be conducted by assessing a random integer from the random integer function and checking to see if it is in range. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.30.7 Tests For

This test validates section 4.4.12, of requirements documentation.

7.31 Splash Rendering Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.31.1 Test Factors Involved

Correctness

7.31.2 Initial State

Game Running State

7.31.3 Inputs

Input: Source image from root folders.

7.31.4 Outputs

Variables are assigned to particular images and have been initialized.

7.31.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.31.6 Methodology

This test will be conducted by assessing the location of the splash for the frame based off coordinates. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.31.7 Tests For

This test validates section 4.4.5, of requirements documentation.

7.32 Audio Sample Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.32.1 Test Factors Involved

Correctness

7.32.2 Initial State

Game Running State

7.32.3 Inputs

Input: Source audio from root folders.

7.32.4 Outputs

Audio output through external device compatible of playing audio.

7.32.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.32.6 Methodology

This test will be conducted by physically assessing the audio output of a swoosh function. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.32.7 Tests For

This test validates section 4.4.15, of requirements documentation.

7.33 Entities Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.33.1 Test Factors Involved

Correctness

7.33.2 Initial State

Game Running State

7.33.3 Inputs

Input: Pipe, Powerup, Fish, Gameover

7.33.4 Outputs

An array of different object needed to display

7.33.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.33.6 Methodology

This test will be conducted by checking to see if the array of entities is not null when different inputs are pushed on it. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.33.7 Tests For

This test validates section 4.4.7, of requirements documentation.

7.34 Canvas Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.34.1 Test Factors Involved

Correctness

7.34.2 Initial State

Game Running State

7.34.3 Inputs

Input: Entities

7.34.4 Outputs

Check to see if canvas is able to display different entities

7.34.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.34.6 Methodology

This test will be conducted by checking to see if each entity is displayed to the canvas. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.34.7 Tests For

This test validates section 4.4.4, of requirements documentation.

7.35 Canvas Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.35.1 Test Factors Involved

Correctness

7.35.2 Initial State

Game Running State

7.35.3 Inputs

Input: Play, Splash, GameOver

7.35.4 Outputs

Change from Play running state to GameOver running state.

7.35.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.35.6 Methodology

This test will be conducted by checking to see if a state is changed through changeState function from Play to GameOver. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.35.7 Tests For

This test validates section 4.4.9-4.4.11, of requirements documentation.

7.36 Update Error Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.36.1 Test Factors Involved

Correctness

7.36.2 Initial State

Game Running State

7.36.3 Inputs

Input: ThrowError with States

7.36.4 Outputs

Catch error and do nothing.

7.36.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.36.6 Methodology

This test will be conducted by checking to see if a null state is changed through and how the game will react. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.36.7 Tests For

This test validates section 4.4.5, of requirements documentation.

7.37 Sound Play Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.37.1 Test Factors Involved

Correctness

7.37.2 Initial State

Game Running State

7.37.3 Inputs

Input: Source sound from root folders.

7.37.4 Outputs

Audio output through external device compatible of playing audio.

7.37.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.37.6 Methodology

This test will be conducted by physically assessing the audio output to check if a sound has been played. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.37.7 Tests For

This test validates section 4.4.15, of requirements documentation.

7.38 Sound Time Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.38.1 Test Factors Involved

Correctness

7.38.2 Initial State

Game Running State

7.38.3 Inputs

Input: Source sound from root folders.

7.38.4 Outputs

Measure the length of sound playing to the designated amount.

7.38.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.38.6 Methodology

This test will be conducted by physically assessing the length of the audio output when a sound has been played.. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.38.7 Tests For

This test validates section 4.4.15, of requirements documentation.

7.39 Animation Frame Testing

This will be a unit test using Jasmine Framework to validate correctness.

7.39.1 Test Factors Involved

Correctness

7.39.2 Initial State

Game Running State

7.39.3 Inputs

Input: Webkit Framework for JavaScript

7.39.4 Outputs

A designated window that is set by the user's browser's dimensions.

7.39.5 Schedule

This test regards the main simulation of the game and therefore will be necessary for the Final Demonstration 1

7.39.6 Methodology

This test will be conducted by physically assessing the size of the application to confirm an appropriate animation frame. It will be conducted through the Jasmine Framework and outputted in our unit test HTML file

7.39.7 Tests For

This test validates section 4.4.3 and 4.4.4, of requirements documentation.