CANICULA CAMPUSUS

TEST SPECIFICATION

Contents

1. INTRODUCTION

The aim of this report is to explain general overview of the Test Specification for the Canicula Campusus. Game's name coming from latin word Canicula which means in English "Dog", and fabrication word Campusus. In this document, we will mention about test plan, unit tests and additional tests. In this way, we will explain project's goal, content and organization.

1.1 Goals

The goal of the project is to develop Android game application with the aid of Unity 3D Platform. This project is game for android users. Our game encapsulate 6 different modes which are chasing cats, daily routine of a dog, running from cars in a traffic in the campus, fastest arrive of a point, daily challenge and multiplayer area wars. One of the most important aim of this project is to provide fun to our game users and have good times with their friends.

The goals of the project team:	• To have created a fun game that keeps player playing.
	• To have created a game that will target users who enjoy playing casual styles mobile gamers.
	• To establish game mechanics that will improve the gameplay in a way that will be appealing to the player.
	• To provide balance between game difficulty level and user experience.

1.2 Contents and Organization of the Document

Organization of this document is as the following;

We will provide testing strategy, test subjects and equivalance partitioning in terms of test model. Moreover, we will prepare test cases and scripts in the unit tests part. Finally, we will explain security testing, performance testing, load/stress testing, acceptance testing, usability testing and functional testing about our project in the additional tests part. Explanation of the content is below:

- **Test Plan:** Test plan is a document detailing a systematic approach to testing a system such as a software project. The plan typically contains a detailed understanding of the eventual workflow.

    - **Testing Strategy:** This part, we will explain testing strategy of our project and we will mention about motivation.
    - **Test Subjects**: In this stage, we will give some information about test subjects in terms of component diagram of our projects. Also we will explain integration testing with the aid of our project's data.
    - **Equivalance Partitioning**: In this phase, we will indicate equivelence partitoning as in the partition valid and invalid input values into equivalance classes.
- **Unit Tests:** Unit testing is an approach to software development in which tests are written for each function in your application.
    - **Test Cases:** This part, we will listed test cases and map them to user stories of our projects.
    - **Scripts:** In this phase, we will explain test cases to show what we are planning to do.
- **Additional Tests:** Finally, we will mention some non-functional tests and functional test in our projects. These tests are listed as below:
    - Security Testing
    - Performance Testing
    - Load/Stress Testing
    - Acceptance Testing
    - Usability Testing
    - Functional Testing

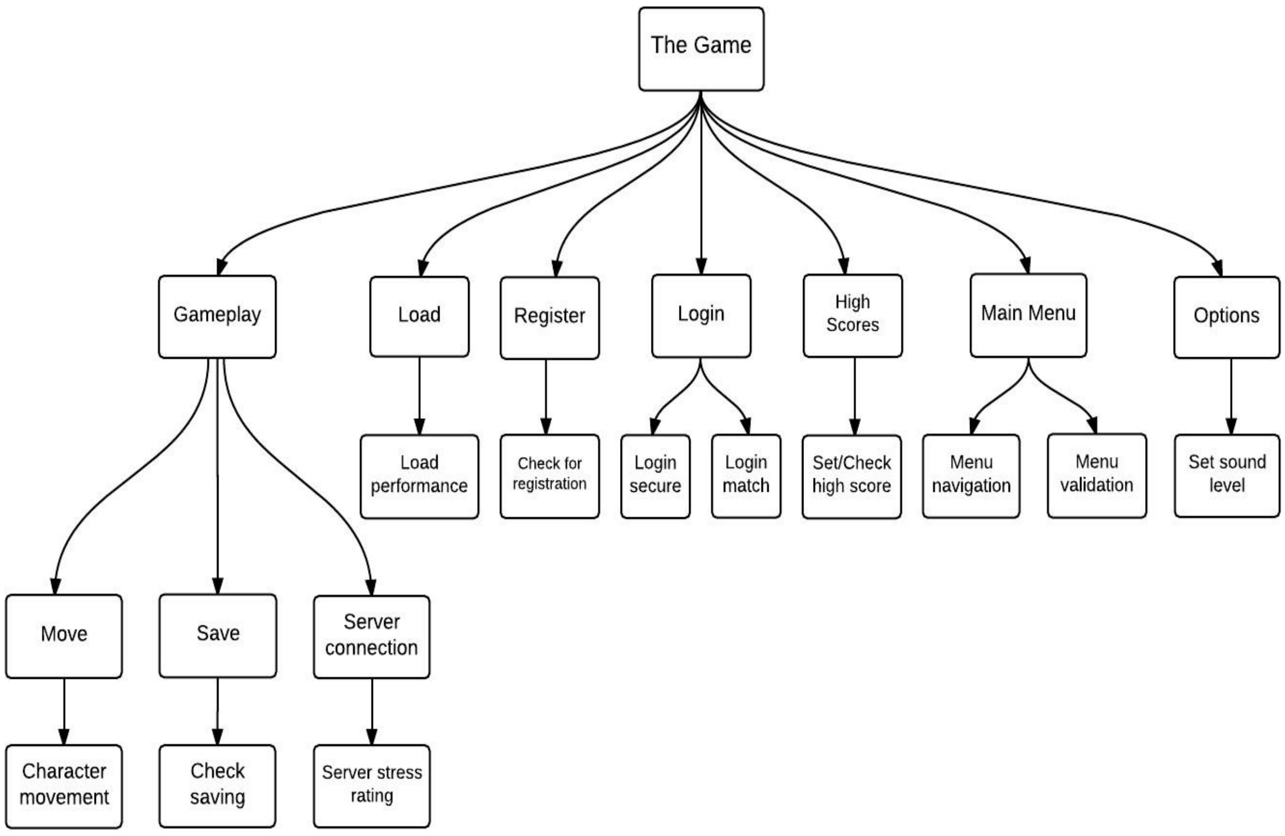
2. TEST PLAN

- Test plan is a document detailing a systematic approach to testing a system such as a software project. The plan typically contains a detailed understanding of the eventual workflow.
- On this wise, we will mention about our project's test strategy, test subjects according to component diagram and equivalance partitioning.

2.1 Testing Strategy

- First of all we decided to test this project with incremental approach because in this approach errors are easy to recognize and fix.
- To considering our project's object oriented structure, the best testing strategy is the bottom-up testing strategy.

- In bottom-up strategy, firstly the lowest layer of the calling hierarchy is tested and then the subsystem which is calling that layer is tested until the test driver's test case routine is ended.
- We use depth-first bottom-up structure for integration test.
- According to our strategy, firstly we will determine suitable test cases for move, save and server components . These components depend on the gameplay.
- After this, other test cases will determined in terms of load, register, login, high scores, main menu and options components.
- All the components depend on the game. Thus, depth-first bottom up strategy is implemented.
- Our project's testing strategy is schematized as below:



<u>Steps:</u>

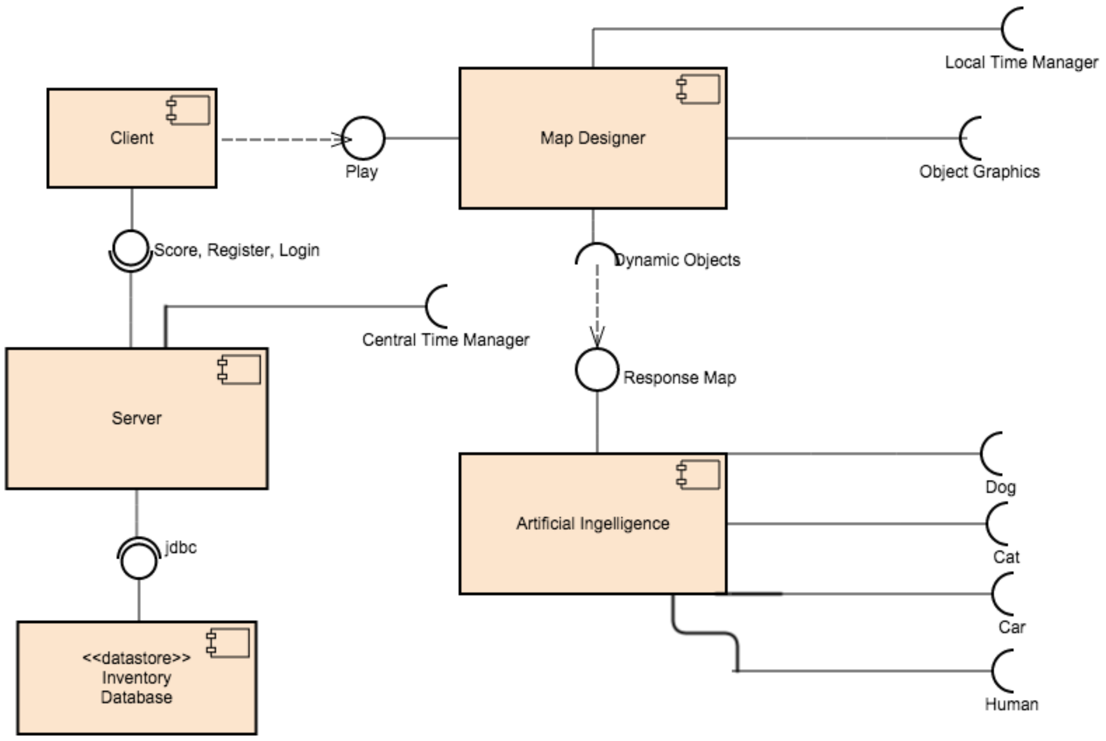
1- Character Movement, Check Saving, Server stress rating

2- Load performance, Check for registration, login structure, login match, set/check high score, menu navigation, menu validation, set sound level, Character Movement, Check Saving, Server stress rating

NOTE: Required mocks are done using mockito and the testing source code files are included in "Test codes.rar" file.

2.2 Test Subjects

- Test subjects will be elaborated according to component diagram.
- Component diagram is as the following:



Component Diagram

- Our Artificial Intelligence Engine needs integration tests because of its included components Dog, Cat, Car and Human; and Map Design Object connections.
- Map Design Object needs integration tests because of its included components Local Time Manager and Object Graphics; and Client Object and Artificial Intelligence Engine connections.
- Client Object needs integration tests because of Map Design Object and Server Object connections.
- Server Object needs integration tests because of its included component Central Time Manager; Client Object and Database connections.

2.3 Equivalence Partitioning

- Our idea behind equivalence partitioning is to divide a set of test conditions into groups.
- In this way, we will categorize equivalence partitioning according to valid and invalid input type:

Input Type	**Inputs**	**Valid Equivalence Classes**	**Invalid Equivalence Classes**
Menu Screen	**Buttons**	Physical buttons (Exit, Volume up, Volume down, Game mode, Login, Register, Cancel, Done, OK etc.)	Touching the screen area that does not cover by any button
EditBoxes	**Keyboard**	Any charatecter on the keyboard except space, tab.	Space and tab characters in the keyboard, empty name and password fields
Game Controls	**Screen Touch**	-Four directions in the screen to move dog. -Any dog in the screen to fight another dog.	Touching neither any dog nor one of the four direction screen area

3. UNIT TESTS

Unit testing is an approach to software development in which tests are written for each function in your application. This part, we will prepare test cases according to user stories and we will mention about scripts which are related to our test cases and test plan.

3.1 Test Cases

- Project's test cases are formed according to user stories.
- Firstly we mentioned user stories of our project. After this, we showed test cases and map them to user stories.
- These stories are created according to behavior-driven development style.
- User stories which are related to unit test are indicated into the box. These stories are listed as the following.
- Map table of user stories and test cases is also shown as below:

Test Case ID	**Test Title**	**Testing Type**	**Mapped User Story**
#01	Login match check	**Unit Test**	Login
#02	Check for Registration	**Unit Test**	Registration
#03	Set and check high score	**Unit Test**	High Score
#04	Character movement	**Unit Test**	Move
#05	Check and test saving mechanism	**Unit Test**	Save
#06	Login security check	Security Test	Login

#07	Load performance rating	Performance Test	Load
#08	Server stress rating	Load/Stress Test	Connect to Server
#09	Validation of Menu	Acceptance Test	Menu
#10	Menu navigation	Usability Test	Menu
#11	Check for background music and sound level	Functional Test	Set Sound Level

Feature: Login

As a user I want to login with my account

Scenario: Player logins to the system

Given players in the Login page

When Player enters the correct user credentials

And Player presses “ok” button

Then player should be logged in to the system

Feature: Register

As a user I want to create a profile

Scenario: player creates a new account

Given Player is in the Register page

When player enters the user credentials

And player press the “ok” button

Then New account should be opened by the system

Feature: Save

As a user I want to save my game to continue later

Scenario: Player saves the current game session

Given Player is in the game

And current game state has changed

When Player opens up the save menu

And Chooses for a suitable slot

Then Current game state should be saved by the system

Feature: High score

As a user I want to score up high score

Scenario: Player beats previous high scrore

Given there is a previous high score

And Player is in the mode with high score

When Player makes a new record in the game

Then new high score should be saved by the system

Feature: Move

As a user I want to move the dog

Scenario: Player makes a movement in the game

Given player is in the game screen

And Dog exist in the screen

When Player touches the screen

Then dog should be moved in desired direction by the system

Test cases are shown as below:

UNIT TEST CASE 1			
Test Case ID	#01	**Test Designed by**	Mert Karaçam
Test Priority	High	**Test Designed date**	
Module Name	Login Screen	**Test Executed by**	Mert Karaçam
Test Title	Login match check	**Test Execution date**	
Description	Test the login for input/output behaivour	**Mapped User Story**	Login

UNIT TEST CASE 2			
Test Case ID	#02	**Test Designed by**	Mert Karaçam
Test Priority	Medium	**Test Designed date**	
Module Name	Saving game	**Test Executed by**	Mert Karaçam
Test Title	Check and test saving mechanism	**Test Execution date**	
Description	Check saving and saved games behaviours	**Mapped User Story**	Save

UNIT TEST CASE 3			
Test Case ID	#03	**Test Designed by**	Mert Karaçam
Test Priority	High	**Test Designed date**	
Module Name	Register Screen	**Test Executed by**	Mert Karaçam
Test Title	Check for Registration	**Test Execution date**	
Description	Creating an account	**Mapped User Story**	Registration

UNIT TEST CASE 4			
Test Case ID	#04	**Test Designed by**	Mert Karaçam
Test Priority	High	**Test Designed date**	
Module Name	Game Screen	**Test Executed by**	Mert Karaçam
Test Title	Character movement	**Test Execution date**	
Description	Moving the character inside the map	**Mapped User Story**	Move

UNIT TEST CASE 5			
Test Case ID	#05	**Test Designed by**	Mert Karaçam
Test Priority	Low	**Test Designed date**	
Module Name	Score function	**Test Executed by**	Mert Karaçam
Test Title	Set and check high score	**Test Execution date**	
Description	Score Calculation and check leader boards	**Mapped User Story**	High Score

3.2 Scripts

Test script in software testing is a set of instructions that will be performed on the system under test to test that the system functions as expected.

These scripts include test number, description, sample input, required mocks and expected result:

Test Title: Login match check			Test Case ID: **1**	
Scenario # : 1		Tester: Mert Karaçam	Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Check textbox for data input	Username and password	Get the inputs from textboxes	Login Function
2	Hide the given password as symbols	"password"	Symbols displayed in textbox	Login Function
3	Check matching for registered user	Correct username and password	Match with registered user	Login Function
4	Click on done button	touch on the done button	User logged in	Login Function
5	Click on cancel button	touch on the cancel button	Back to the menu	Login Function

Test Title: Check and test saving mechanism				Test Case ID: **2**
Scenario # : 2		Tester: Mert Karaçam		Date of Test:
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Check if the game state can be saved	New game state to be saved	New saved slot in the saved games	Save Function
2	Check saving works with overwriting	New game state to be saved after all slots are full	New game state saved on another existing slot	Save Function
4	Check name and date informations on save slots	"Suitable name and auto date"	Display correct name and date on save slots	Save Function
5	Click on done button	touch on the done button	Game state saved	Save Function
6	Click on cancel button	touch on the cancel button	Back to the game	Save Function

Test Title: Check for Registration				Test Case ID: **3**
Scenario # : 3		Tester: Mert Karaçam		Date of Test:
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Length of the password	Three or less character password	Error for a short password	Register Function
2	Length of the password	Twelve or more character password	Error for a long password	Register Function
3	Length of the username	Five or less character username	Error for a short username	Register Function
4	Length of the username	Twelve or more character username	Error for a long username	Register Function
5	Check for an existing username	"Username"	Error for existing username	Register Function

Test Title: Character movement			Test Case ID: **4**	
Scenario # : 4	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Moving in different directions	Touches around the character	Character appears in new location	Move function
2	Check movement outside of screen	Touches towards edge of the screen	Character stays in the screen	Move function
3	Check movements through the building	Touches to move through the building	Stays out of the building (collision zone)	Move function

Test Title: Set and check high score			Test Case ID:**5**	
Scenario # : 5	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Setting a new high score	Get a score higher than high score	Score is set as new high score	Score function
2	Checking high scores	Touches on the high score button	High scores are displayed	Score function

NOTE: According to gherkin format, user stories are done using cucumber and the testing source code files are included in “Test codes.rar” file.

4. ADDITIONAL TESTS

Test Case ID	**Test Title**	**Testing Type**	**Mapped User Story**
#01	Login match check	Unit Test	Login
#02	Check for Registration	Unit Test	Registration
#03	Set and check high score	Unit Test	High Score
#04	Character movement	Unit Test	Move
#05	Check and test saving mechanism	Unit Test	Save
#06	Login security check	**Security Test**	Login
#07	Load performance rating	**Performance Test**	Load
#08	Server stress rating	**Load/Stress Test**	Connect to Server
#09	Validation of menu	**Acceptance Test**	Menu
#10	Menu navigation	**Usability Test**	Menu
#11	Check for background music and sound level	**Functional Test**	Set Sound Level

4.1 Security Testing

Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended.

Veracode tool is used for security testing. Our project's security test is itemized and test case and user story are given as the following:

- The security test is applied on the login module in order to check whether the login module allows registered users to access the game when they enter correct user name and password and to check whether the module prevents accesses with incorrect user names or passwords.
- The security test is applied on the login module by entering invalid user name and password on the login page and by entering valid user name and password in another check.
- The response of the system to each of the valid and invalid inputs is observed and checked with the expected results.

Related User Stories:

Feature: Login

As a user I want to login with my account

Scenario: Player logins to the system

Given players in the Login page

When Player enters the correct user credentials

And Player presses “ok” button

Then player should be logged in to the system

SECURITY TEST CASE			
Test Case ID	#06	Test Designed by	Mert Karaçam
Test Priority	High	Test Designed date	
Module Name	Login Screen	Test Executed by	Mert Karaçam
Test Title	Login security check	Test Execution date	
Description	Test the login page	Mapped User Story	Login

Scenario # : 6		Tester: Mert Karaçam	Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Enter the wrong user name	“wrong user name”	Error message	Login Function
2	Enter the wrong user password	“wrong user password”	Error message	Login Function
3	Repeat the first two steps	For two times wrong password	15 minutes ban to username	Login Function
4	Hide the given password as symbols	“password”	Symbols displayed in textbox	Login Function

4.2 Performance Testing

Performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload.

WAPT tool is used for performance testing. Our project's performance test is itemized and test case and user story are given as the following:

- The performance test is applied on the load module in order to check whether the load module succeses loading an existing saved game and to check whether the game is loaded in an acceptable amount of time.
- The performance test is applied on the load module by choosing an existing saved game in the load page.
- The procedure is applied by touching where the data for the existing saved session is located on the screen in the load page.
- Then, it is checked whether the selected saved session is loaded by the system.
- If loading is successful, the duration of the load operation is observed and checked with the expected result.

Related User Story:

Feature: Load

As a user I want to load a saved game

Scenario: Player loads previously saved game

Given player is in the Load page

And there is a saved session in the list

When Player chooses the saved session

And Player press the "ok" button

Then desired session should be loaded by the system

PERFORMANCE TEST CASE			
Test Case ID	#07	**Test Designed by**	Mert Karaçam
Test Priority	Medium	**Test Designed date**	
Module Name	Load	**Test Executed by**	Mert Karaçam
Test Title	Load performance rating	**Test Execution date**	
Description	Test the loading function	**Mapped User Story**	Load

Scenario # : 7		Tester: Mert Karaçam		Date of Test: 10/12/2014
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Clicks on the saved game	Screen touch	Chosen saved game will be loaded	Load function
2	Evaluate loading time		Loading function complexity	Load function

4.3 Load/Stress Testing

WAPT tool is used for load testing. Our project's load test is itemized and test case and user story are given as the following:

- The load/stress test is applied on the server module in order to check whether the server module responses users without much delay when the server is in high load, that is, when the number of active users in the server is high.
- The load/stress test is applied on the server module by running the server in high load.
- Several operations are tried in the server such as connecting to a game, creating a new game and making movements in the game while the server is running in high load.
- The response of the server to each of the applied operations are observed and checked with the expected results.
- The response time of the server to each operation is also checked with the expected results.

Related User Story:

Feature: Connect to Server

As a user I want to play multiplayer game

Scenario: Player connects multiplayer game

Given player logged in

And player is connected to the inernet

And player is in the lobby screen

When player chooses the server

And player press the "ok" button

Then Connection should be supply by the system

LOAD/STRESS TEST CASE			
Test Case ID	#08	**Test Designed by**	Mert Karaçam
Test Priority	Medium	**Test Designed date**	
Module Name	Server connection	**Test Executed by**	Mert Karaçam
Test Title	Server stress rating	**Test Execution date**	
Description	Test the connection function	**Mapped User Story**	Connect to Server

Scenario # : 8	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Connect to server with multiple account	High number of connections	Keeping the same response time before	Connection function
2	Evaluate response time		Average server response time	Connection function

4.4 Acceptance Testing

TestPlan tool is used for acceptance testing. Our project's acceptance test is itemized and test case and user story are given as the following:

- The user acceptance test is applied on the gameplay module in order to check whether users can play in any of the modes of the game.
- The user acceptance test is applied on the gameplay module by trying to create a new game in each of the available game modes.
- The user acceptance test is applied on the exit button so as to close the game.

Related User Story:

Feature: Menu

As a user I want to validate between menu buttons

Scenario: Player clicks on desired button

Given player is in the menu

And Menu buttons are displayed

When Player touches a button

Then expected screen should be shown by the system

ACCEPTANCE TEST CASE			
Test Case ID	#09	**Test Designed by**	Mert Karaçam
Test Priority	High	**Test Designed date**	
Module Name	Game Menu	**Test Executed by**	Mert Karaçam
Test Title	Validation of menu	**Test Execution date**	
Description	Test the acceptance and validation of menu	**Mapped User Story**	Menu

Scenario # : 9	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Touch on each buttons	Touches on the screen	Expected screen displayed	Menu validation
2	Checking exit button	Touch on the exit button	Close the game	Menu validation
3	Game mode buttons open the specific mode	Touches on the specific game mode buttons	Start the specific game modes	Menu validation

4.5 Usability Testing

APPLause tool is used for usability testing. Our project's usability test is itemized and test case and user story are given as the following:

- The usability test is applied on the menu module in order to check whether the users can use the menu easily and learn and understand the features of the menu buttons quickly.
- The usability test is applied on the menu module by letting the volunteer users use the menu and by getting feedbacks about their experiences of using the menu buttons.
- The feedbacks from the users are collected and obtained feedbacks are compared with the expected results.

Related User Story:

Feature: Menu

As a user I want to navigate between menu buttons

Scenario: Player clicks on desired button

Given player is in the menu

And Menu buttons are displayed

When Player touches a button

Then expected screen should be shown by the system

USABILITY TEST CASE			
Test Case ID	#10	**Test Designed by**	Mert Karaçam
Test Priority	High	**Test Designed date**	
Module Name	Game Menu	**Test Executed by**	Mert Karaçam
Test Title	Menu navigation	**Test Execution date**	
Description	Test the usability of menu	**Mapped User Story**	Menu navigation

Scenario # : 10	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Let people see shape of the menu buttons	Different user types	Good opinions on the button shapes	Menu navigation
2	Colors displayed in the menu	Different user types	Good opinions on the menu colors	Menu navigation
3	Let people see size of the menu buttons	Different user types	Good opinions on the button sizes	Menu navigation
4	location of the menu buttons inside the screen	Different user types	Good opinions on the button locations	Menu navigation

4.6 Functional Testing

MonkeyTalk tool is used for functional testing. Our project's functional test is itemized and test case and user story are given as the following:

- The functional test is applied on the sound module in order to check whether the sound module adjusts the sound of the game correctly when the user changes the sound level from the options page.
- The functional test is applied on the sound module by selecting different sound levels from the options page.
- For each selected sound level, it is checked whether the system adjusts the sound of the game correctly.
- The responses of the system to the selections of different sound levels are checked with the expected results.

Related User Story:

Feature: Set sound level

As a user I want adjust sound level

Scenario: Player changes sound level of the game

Given player is in the Options page

When Player adjusted the sound level bar as desired

And Player press the "done" button

Then sound level should be changed by the system

FUNCTIONAL TEST CASE			
Test Case ID	#11	**Test Designed by**	Mert Karaçam
Test Priority	Low	**Test Designed date**	
Module Name	Settings	**Test Executed by**	Mert Karaçam
Test Title	Check for background music and sound level	**Test Execution date**	
Description	ON/OFF sound & background music	**Mapped User Story**	Set Sound Level

Scenario # : 11	Tester: Mert Karaçam		Date of Test:	
Test Number	Description	Sample Input	Expected Result	Required Mock
1	Setting different sound levels	Adjusted sound level	Changing of sound levels	Sound adjusting function
2	Setting On or Off menu music	Touches on the on/off button	On or Off menu sound	Sound adjusting function

NOTE: According to gherkin format, user stories are done using cucumber and the testing source code files are included in “Test codes.rar” file.