**REVISION HISTORY**

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 20.10.2024 | 1.0 | Added Tests Preparations | Ömer Emre Bozkurt, D. Yağmur Uğut |
| 25.10.2024 | 1.1 | Added specific tests for SRS-REQ-001 and SRS-REQ-002 | Ömer Emre Bozkurt |
| 26.10.2024 | 1.2 | Added specific tests for SRS-REQ-005 and SRS-REQ-010 | Doruk Esen |
| 27.10.2024 | 1.3 | Added specific tests for SRS-REQ-008 and SRS-REQ-011 | D. Yağmur Uğut |
| 27.10.2024 | 1.4 | Added specific tests for SRS-REQ-003 and SRS-REQ-004 | Poyraz KÖROĞLU |
| 28.10.2024 | 1.5 | Added specific tests for  SRS-REQ-006 and SRS-REQ-009 | Eda Nur Yılmaz |
| 02.11.2024 | 1.6 | Modified tests, added SRS-REQ-008.1, SRS-REQ-008.2, and SRS-REQ-0011.1 | D. Yağmur Uğut |
| 02.11.2024 | 1.7 | Modified test,, added SRS-REQ001.1 and SRS-REQ-001.2 | Poyraz KÖROĞLU |
| 02.11.2024 | 1.8 | Added specific test for SRS-REQ-001.2 | D. Yağmur Uğut |
| 03.11.2024 | 1.9 | Added specific test for SRS-REQ-006.1 and updated test T-SRS-REQ-001 for SRS-REQ-005.2 | Ömer Emre Bozkurt |
| 03.11.2024 | 2.0 | Modified tests for SRS-REQ-005 and SRS-REQ-010  Added specific tests for SRS–REQ-001.4 and SRS-SB-002.1 | Doruk Esen |
| 03.11.2024 | 2.1 | Modified tests for SRS-REQ-006 and SRS-REQ-009.  Added tests for SRS-REQ-002.2 and SRS-REQ-003.1. | Eda Nur Yılmaz |
| 04.11.2024 | 2.2 | Updated existing test cases for clarity and accuracy. New test cases were added to cover additional requirements. Renumbered all test cases in ascending order for improved organization. Enhanced documentation for all test cases. | Doruk Esen  D. Yağmur Uğut  Eda Nur Yılmaz  Poyraz Köroğlu  Ömer Emre Bozkurt |

**TABLE OF CONTENTS**

[1 Introduction 3](#_heading=)

[1.1 Document overview 3](#_heading=h.1fob9te)

[1.2 References 3](#_heading=)

[1.3 Conventions 3](#_heading=h.2et92p0)

[2 Tests preparations 4](#_heading=)

[2.1 Hardware preparation 4](#_heading=)

[To create the game we have imagined, we have to have the necessary equipment to be able to work as a team simultaneously. That is why we have decided to work with our most capable devices. 4](#_heading=)

[2.2 Software preparation 4](#_heading=)

[2.3 Other test preparation 4](#_heading=)

[2.4 Safety, security and privacy precautions 4](#_heading=)

[3 Tests descriptions 5](#_heading=)

[3.1 Software Tests for Requirements 5](#_heading=)

# **Introduction**

## ***Document overview***

This document is the software test plan of the SB software development project. It contains the description of tests.

## ***References***

| # | Document Identifier | Document Title |
| --- | --- | --- |
| [SRS] | SB-SRS-1 | SB Software Requirements Specifications |
|  |  |  |

## ***Conventions***

Each test case will be presented in a table format as shown below:

| **Test ID** | <ID of the test> |  |
| --- | --- | --- |
| Test description | <A brief description of the test> |  |
| Verified Requirement | <ID numbers of requirements that are to be verified with this test.> | Verification method: <I,A,D,T> |
| Initial conditions | <The state of software before test> |  |
| Tests inputs | <Input data from any test tool, input files name and location, if any > |  |
| Data collection actions | <Recording and post processing of output data, if any > |  |
| Tests outputs | <Output data files, if any> |  |
| Assumptions and constraints | If any, may be limited access to a tool, license … |  |
| Expected results and criteria | <list of expected results> | <special criteria to be met by the result, if any> |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and evaluation criteria** |
| 1 | <actions listed row by row> | <expectation after each action if any> |
| 2 | … | … |

# **Tests preparations**

This section contains tasks and recommendations before executing tests.

## ***Hardware preparation***

## To create the game we have imagined, we have to have the necessary equipment to be able to work as a team simultaneously. That is why we have decided to work with our most capable devices.

**Our Devices:**

* Monster Tulpar T5 V19.2 RAM:32GB, GPU: Nvidia RTX2060, CPU: Intel Core i7-10750H, SSD:256GB, HDD: -

* Huawei Matebook D16, RAM:16GB, GPU: Intel Iris X Graphics, CPU: Intel Core i7-12700H, SSD:512GB, HDD: -

* MacBook Air M1: RAM:8GB, CPU: Apple M1, SSD: 256GB, HDD-, Speaker: Stereo

* HP Victus 16-R1050NT, RAM:32GB, GPU: Nvidia RTX 4050, CPU: Intel Core i7-14700HX, SSD: 1TB, HDD: -

* MSI Prestige 14 Evo B13M, RAM: 32 GB, GPU: Intel Evo Graphics, CPU: Intel Core i7-13700H 2.40 GHz, SSD: 1TB, HDD: -

## ***Software preparation***

Testing Framework: JUnit 5.7.1  
Version control: Github

## ***Other test preparation***

None.

## ***Safety, security and privacy precautions***

None.

# **Tests descriptions**

## Software Tests for Requirements

| **Test ID** | **T-SRS-REQ-001** |  |
| --- | --- | --- |
| Test desc. | Verify that when the game is over, the game saves and displays if it is in the top 5 high scores. |  |
| Verif. Req. | SRS-REQ-001 | Demonstration |
| Init. Cond. | SB Software is started and played game |  |
| Tests inputs | N/A |  |
| Data collection | Save files of high scores |  |
| Tests outputs | top 5 high scores from previous games |  |
| Assum & constr | N/A |  |
| Expected results and criteria | A high score file should be generated previously on the same location with software | The save file contains scores of previously played games. |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The user starts SB Software. | The software reads high scores from previously played games and lists top 5 scores on the main menu |
| 2 | The user can see top 5 high score o | Software displays top 5 high scores |

| **Test ID** | **T-SRS-REQ-002** |  |
| --- | --- | --- |
| Test desc. | Verify that the SB software allows users to click the “Start Game” button and start a new game, spawning aliens in a predetermined pattern. |  |
| Verif. Req. | SRS-REQ-002, SRS-REQ-002.1 | Demonstration |
| Init. Cond. | SB Software is started |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | N/A |  |
| Expected results and criteria | SB Software is on the Main Screen |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | SB software is on the Main Screen | SB software displays “Start Game”, and “Quit” buttons, and top 5 high scores |
| 2 | The user clicks the “Start Game” Button | SB software displays the game screen, spa |
| 3 | SB software switches to Game Screen and starts a new game. | The new game starts, the alien set appears on the screen in a predetermined pattern and UI elements appear on the game screen |
| 4 | The movement pattern of the spawned enemies is observed. | Enemies follow the expected pattern consistently without deviation. |
| 5 | The game progresses further until a new wave spawns. | A new wave of enemies appears and follows its designated movement pattern. |
| 6 | The user repeats the game up to this stage to verify pattern consistency. | Patterns remain consistent and predictable in each test run, ensuring predetermined movement. |

| **Test ID** | **T-SRS-REQ-003** |  |
| --- | --- | --- |
| Test desc. | Verify that players can move the paddle to the left and right side of the screen with the left and right arrow keys. |  |
| Verif. Req. | SRS-REQ-003, SRS-REQ-003.1 | Test |
| Init. Cond. | SB software has started and the paddle stays still in the bottom middle of the screen |  |
| Tests inputs | N/A |  |
| Data collection | Record that the paddle moves according to the arrow key presses (left or right) |  |
| Tests outputs | N/A. |  |
| Assum & constr | Testing is conducted on a desktop environment with a keyboard. |  |
| Expected results and criteria | The paddle moves right when the right arrow key is pressed and left when the left arrow key is pressed. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The operator presses and holds the right arrow key. | The paddle moves right until the key is released or the rightmost end of the paddle hits the end of the game tab. |
| 2 | The operator presses and holds the left arrow key. | The paddle moves left until the key is released or the left-most end of the paddle hits the end of the game tab. |

| **Test ID** | **T-SRS-REQ-004** |  |
| --- | --- | --- |
| Test desc. | Verify that the ball doesn't go through the paddle stationed at the middle bottom of the screen and bounces instead. |  |
| Verif. Req. | SRS-REQ-004, SRS-REQ-004.1 | Test |
| Init. Cond. | SB software has started and the ball, paddle, and aliens are visible on the screen once the game has started. |  |
| Tests inputs | N/A |  |
| Data collection | Record that the ball bounces when it comes in contact with the paddle and hits an alien causing the alien damage. |  |
| Tests outputs | N/A. |  |
| Assum & constr | Testing is conducted on a desktop environment with a keyboard. |  |
| Expected results and criteria | The ball bounces from the paddle while both stationary and moving.  The alien is damaged when in contact with the ball |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | Start the game, wait for the ball to release, and come into contact with the paddle. | The ball bounces from the paddle without the user pressing any keys. |
| 2 | After the initial bounce move the paddle to hit the ball again while the paddle is moving. | The ball bounces from the paddle while the paddle is moving. |
| 3 | Bouncing from the paddle the balls come in contact with an alien. | The alien is damaged when in contact with the ball |

| **Test ID** | **T-SRS-REQ-005** |  |
| --- | --- | --- |
| Test desc. | Verify that random power-ups appear after destruction of some enemies and those power ups grant additional buffs to the player |  |
| Verif. Req. | SRS-REQ-005, SRS-REQ-005.1, SRS-REQ-005.2 | Test |
| Init. Cond. | SB software has started and the player kills some enemies. |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | The test is conducted on a desktop environment with a keyboard.  The game should be played for a few rounds to allow power-up aliens to spawn. |  |
| Expected results and criteria | When a booster is hit with the ball, the user earns the booster’s effect. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The user opens the game. | Aliens start to generate and move towards the user's paddle; new waves appear as previous waves are defeated. |
| 2 | The player continues playing the game. | Power-up aliens with unique appearances should spawn alongside regular aliens and continue to appear as gameplay progresses. |
| 3 | The user kills a power-up alien. | The power-up is activated, granting the user the specified effect. |
| 4 | If the user has no active boosters, the booster takes effect immediately. | Booster’s name is displayed on the bottom left corner of the screen. |
| 5 | If there are already active boosters, only non-conflicting boosters are activated concurrently | New booster names are added to the list at the bottom left corner. |
| 6 | If a new booster conflicts with an active one, the booster with the higher remaining time becomes active. | The longer-lasting booster remains active. |

| **Test ID** | **T-SRS-REQ-006** |  |
| --- | --- | --- |
| Test desc. | Verify that the SB software can take input from the user to save it with their score for the best scores list. |  |
| Verif. Req. | SRS-REQ-006, SRS-REQ-006.1 | Test |
| Init. Cond. | SB Software is started  Game has ended |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | The user scores higher than or the same as one of the previous top 5 scores. If all the same, it should be displayed in alphabetical order. |  |
| Expected results and criteria | High scores should match the names given by each user and save them to the high scores list. | See below |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The game is lost by the user. | Message displayed “Game Over” and  “Enter Name”. The name input area should open if the score is in the top 5 high scores list. |
| 2 | The user writes a three-character name. | The written name should be displayed on the screen. SB software matches names with high scores and saves them to the high-score file. |
| 3 | The game is opened by the user again. | When the game is opened again, the high scores list on the main screen should contain the name and the score. |

| **Test ID** | **T-SRS-REQ-007** |  |
| --- | --- | --- |
| Test desc. | Verify that SB Software allows players to quit the game by pressing the “Quit” button. |  |
| Verif. Req. | SRS-REQ-007 | Demonstration |
| Init. Cond. | SB Software is started, and the game is active. |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | N/A |  |
| Expected results and criteria | The game closes when the “Quit” button is pressed. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The player clicks on the “Quit” button. | The game closes immediately. |

| **Test ID** | **T-SRS-REQ-008** |  |
| --- | --- | --- |
| Test desc. | Verify that players lose a life if they fail to hit the ball with the paddle, causing it to fall below it. |  |
| Verif. Req. | SRS-REQ-008, SRS- REQ-008.1, SRS-REQ-008.2 | Test |
| Init. Cond. | SB software has started and the player has lives remaining. |  |
| Tests inputs | N/A |  |
| Data collection | Record the player’s remaining lives before and after the ball falls below the paddle.  Capture any game logs indicating the life decrement event. |  |
| Tests outputs | N/A |  |
| Assum & constr | Testing is conducted on a desktop environment with a keyboard |  |
| Expected results and criteria | The player’s life count should decrease by 1 when the ball falls below the paddle.  The game should display a “Game Over” screen if the player has zero lives. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | Launch the game | The game starts with the player’s initial life count displayed. |
| 2 | Allow the ball to fall below the paddle without hitting it. | The player’s life count decreases by 1. |
| 3 | Repeat the action until the player has 0 lives. | The “Game Over” screen appears when all lives are lost. |

| **Test ID** | **T-SRS-REQ-009** |  |
| --- | --- | --- |
| Test desc. | Verify that when the game starts, the player is born with 3 initial lives. |  |
| Verif. Req. | SRS-REQ-009 | Test |
| Init. Cond. | SB Software is started |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | The game should be newly started, without progress. |  |
| Expected results and criteria | The lives in the game screen should contain three lives numerically. | See below |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The game is started by the user. | The game screen opens. The game starts and the user is given 3 lives. UI displays the number of lives that you start with as three. |
| 2 | The user plays the game and loses balls/lives by not being able to stop the ball from falling off from the bottom part of the game screen. | The lives of the users decrease as they lose balls. UI gets updated. |
| 3 | The user loses the game and has no lives left in the end. | Game Over screen should open.  When the game ends, the user should have 0 lives. |

| **Test ID** | **T-SRS-REQ-010** |  |
| --- | --- | --- |
| Test desc. | Verify that players can see their remaining lives, current score, number of lives remaining, active power-ups, current wave or level on the screen while they are playing |  |
| Verif. Req. | SRS-REQ-010, SRS-REQ-010.1 |  |
| Init. Cond. | The game has started.  Total lives are set as 3 or updated as the game progresses. |  |
| Tests inputs | N/A |  |
| Data collection | Record the change of lives and score by either a loss or a surplus |  |
| Tests outputs | N/A |  |
| Assum & constr | The test is being conducted on a regular screen.  No object overlaps with the window that shows total lives, score and other information. |  |
| Expected results and criteria | All information is displayed in a clear and consistent part of the screen.  Total number of lives changes simultaneously depending on the condition.  Score increases simultaneously when an alien is killed.  The total number of lives, current score and wave/level should be visible at all times. Active power-ups should be visible when gained. |  |
| **Test Procedure** |  |  |
| **Step Number** | **Operator Actions** | **Expected result and eval crit** |
| 1 | User launch the game | The starting number of lives should be 3 and be seen clearly. Also, current score and wave/level should be seen |
| 2 | Begin playing and reach a specific point threshold | The total amount of lives should increase by one and screen updates simultaneously |
| 3 | While playing kill an alien with power-up | Power-up that earned should be seen with duration |
| 4 | While playing the user fails to hit the ball | The total amount of lives should decrease by one and screen updates simultaneously |
| 5 | While the user has no life, the user fails to hit the ball | The game ends |

| **Test ID** | **T-SRS-REQ-011** |  |
| --- | --- | --- |
| Test desc. | Verify that players can see the current score displayed on the screen while they are playing and that the game calculates the score based on the type of enemies destroyed and any active score multipliers. |  |
| Verif. Req. | SRS-REQ-011, SRS-REQ-011.1 | Test |
| Init. Cond. | The game is launched and the play screen is active.  The score is initialized to zero or a starting value as per game design. |  |
| Tests inputs | N/A |  |
| Data collection | Record the visibility and location of the score on the screen.  Capture screenshots to verify the score display.  Log score changes when different types of enemies are destroyed and when score multipliers are active. |  |
| Tests outputs | N/A |  |
| Assum & constr | Testing on a standard display setup.  The screen should not have any overlays or obstructions that would hide the score.  Various enemy types and score multipliers are available for the player to encounter during gameplay. |  |
| Expected results and criteria | The current score is displayed on the screen in a visible and consistent position.  The score updates in real-time as the player progresses through the game.  The score increments correctly based on the type of enemies destroyed and any active score multipliers.  The score display remains visible and unobstructed throughout gameplay. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | Launch the game. | The score should be visible on the screen at the start and it should be 0. |
| 2 | Begin playing and destroy a basic enemy. | The score should increase according to the point value assigned to basic enemies, and it updates in real time on the screen. |
| 3 | Destroy a higher-value enemy | The score should increase by a greater amount, reflecting the higher point value. |
| 4 | Activate a score multiplier (if available) and destroy an enemy. | The score increment should reflect the multiplier effect (e.g., double points). |
| 5 | Continue playing to verify that the score remains visible and updates correctly throughout gameplay. | The score display remains visible, and updates accurately based on actions, until the end of the session. |

| **Test ID** | **T-SRS-REQ-012** |  |
| --- | --- | --- |
| Test desc. | Verify that SB Software allows players to return to the main menu after the game ends. |  |
| Verif. Req. | SRS-SB-012 | Test |
| Init. Cond. | SB Software is started, and the game has ended. |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | N/A |  |
| Expected results and criteria | The main menu is displayed when the user selects the option to return to the main menu after the game ends. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The game ends, and the player selects the “Return to Main Menu” option. | The main menu appears on the screen, allowing the player to start a new game or access other menu options. |

| **Test ID** | **T-SRS-REQ-013** |  |
| --- | --- | --- |
| Test desc. | The game shall feature four types of enemies, each with different health points and score values. When an alien’s health score has decreased, it should change disguise. |  |
| Verif. Req. | **SRS-REQ-013, SRS-REQ-013.1** | Test |
| Init. Cond. | SB Software is started |  |
| Tests inputs | N/A |  |
| Data collection | N/A |  |
| Tests outputs | N/A |  |
| Assum & constr | The user should harm/kill all types of aliens. |  |
| Expected results and criteria | The game should generate all types of aliens according to the difficulty level/ how much the game has progressed. | See below |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | The game is started by the user. | The game screen opens. A set of aliens with different disguises is generated. |
| 2 | The user plays the game and harms different types of aliens. | The aliens change disguises according to their harm level. (For a lower amount of health, a specific disguise, and for a bigger amount of health a different disguise.) |
| 3 | The user kills an alien with a disguise specific for ones with the lowest health level. | The alien should disappear from the screen. |
| 4 | The user should continue playing the game and as it gradually starts getting difficult, they should verify if all types of aliens are generated occasionally. | The game generates different types of aliens according to their probability percentage and they should appear occasionally as the game progresses. |

| **Test ID** | **T-SRS-REQ-014** |  |
| --- | --- | --- |
| Test desc. | Verify that the SB Software increases difficulty over time by adjusting enemy speed, spawn rate, and adding more powerful enemies as the player progresses. |  |
| Verif. Req. | SRS-SB-014 | Test |
| Init. Cond. | SB Software has started, and the game is in progress. |  |
| Tests inputs | N/A |  |
| Data collection | Log the difficulty level and enemy attributes over time as the game progresses. |  |
| Tests outputs | N/A |  |
| Assum & constr | Difficulty adjustment should be gradual and noticeable as the game progresses. |  |
| Expected results and criteria | Enemy speed, spawn rate, and strength should increase over time, creating a progressive challenge for the player. |  |
| **Test procedure** |  |  |
| **Step number** | **Operator actions** | **Expected result and eval crit** |
| 1 | Start a new game session and play for an extended period. | Observe that enemy speed and spawn rate increase over time. |
| 2 | Progress to a higher level or time threshold. | More powerful enemies appear, and difficulty escalates accordingly. |