Software Implementation and Testing Document

For

Group 14

Version 1.0

Authors:
Emmanuel Ayala (Student ID: eoa18, GitHub ID: Eoa18)
Ashton Frias (Student ID: agf19e, GitHub ID: AshtonFrias)
Jordan Locke (Student ID: jll19g, GitHub ID: jll19gGH)
Justin Mulder (Student ID: jtm18j, GitHub ID: millingmadly)

1. Programming Languages (5 points)

Since we are using Unity, we are programming in C#.

2. Platforms, APIs, Databases, and other technologies used (5 points)

As of now, we are just using Unity and its built-in features.

3. Execution-based Functional Testing (10 points)

We all have been contributing to adding the functional requirements. For example, in this increment, I (Jordan Locke) added a health bar, Ashton Frias improved the character movement, Justin Mulder added obstacles, and Emmanuel Ayala added enemies. Whatever requirement we are working on, we test it on our own before pushing. If any bugs are discovered after pushing, we let everyone know through Discord and whoever was working on it fixes it then re-pushes. Since we are adding elements of the game piece-by-piece, we usually test our feature through manipulating the functions to imitate something happening even if it actually isn't. For instance, I (Jordan Locke) added a health bar prior to an enemy or obstacle existing in the game. In order to test the health functionality (i.e. losing health), I manually coded the hamster taking damage. Once the enemies and obstacles were added, it was tested again but with all the elements now present.

4. Execution-based Non-Functional Testing (10 points)

For the non-functional requirements, just like for the functional testing, we each test our code piece by piece. We will be able to test for the non-functional requirements more efficiently once the game is complete and all of the elements are in-game.

5. Non-Execution-based Testing (10 points)

We mainly have been relying on execution-based testing because we have not had many issues so far. There was one issue with importing assets (images pulled from GitHub were disproportionate in-game) where we used a mix of non-execution-based testing and execution-based testing because the code was reviewed through Discord before using execution to find the issue.