Software Implementation and Testing Document

For

Group “Riskier”

Version 2.0

Authors:
Bradley Hodson
Michael Styron
Antonio Vidal
Grayson Wagstaff
Wesley Watkins

1. Programming Languages (5 points)

a. C#
    i. C# will be used for scripting within Unity. C# is the main scripting language within Unity and can be used to customize and improve upon the "scenes" within Unity.
b. Python 3.6
    i. Python will be used in the Game AI portion. Primarily Unity would call the AI (created with TensorFlow) as a standalone application and feed in the factors of the game to create an output.

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

a. Unity Game Engine (https://unity.com/)
    i. Unity is the main platform that we will build our game on. Unity is an industry leading game engine that provides a robust interface for creating and customizing games.
b. TensorFlow (https://www.tensorflow.org/)
    i. For our current plan we plan on using TensorFlow to build a "smart" enemy within our game. One member is dedicated to training the AI and learning how to implement it to the game as needed.
c. Google Firebase (https://firebase.google.com/)
    i. As a stretch goal we hope to implement Google Firebase as leaderboard system and allow users to login to track their game statistics.

3. Execution-based Functional Testing (10 points)

a. In the project, before a branch is merged into the master, two reviewers must test and document each set requirement for that branch. This will test if the implementer has met the requirements or if there are additional changes that need to be made. Although this will be at the end of sprint, implementers will attempt to submit early enough to still do revisions before the final day of the sprint.

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

a. During the project we plan to use automated pipelines within GitHub & Unity to pass build checks. By using these automated pipelines, we will test if the program will compile and alert the programmer if certain requirements are met (such as syntax and ability to execute).

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

a. If we plan to review the main board and fighting game code together as a group. While code reviews are not specified as part of the review process, by reviewing the code as a group we will be able to give constructive criticism and learn portions of the program together.