**Software Implementation and Testing Document**

**For**

**Group 2**

Version 1.0

**Authors**:

Zachary Ortiz

Rafe

Joe

# Programming Languages (5 points)

We have been using GDScript for programming the player controls, as Godot and C++ have not been as easily compatible as we’d like. (Z)

C# support is also only available on a separate version of the engine, which is another reason that GDScript was chosen. (J)

*List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).*

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

The game is being built in Godot, a game designer engine. (Z)

Godot supports its own native language, GDScript, which is fairly similar to C#. This is what the game’s code will be written in.

*List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).*

# Execution-based Functional Testing (10 points)

For Mario’s movements and player controls, Zach has tested it on numerous occasions in Godot’s debugger. (Z)

For enemy behavior, Joe has created a simple “player” and was able to test enemy functionality, animations, and death behavior in Godot’s debugger. (J)

*Describe how/if you performed functional testing for your project (i.e., tested for the* ***functional requirements*** *listed in your RD).*

# Execution-based Non-Functional Testing (10 points)

*Describe how/if you performed non-functional testing for your project (i.e., tested for the* ***non-functional requirements*** *listed in your RD).*

# Non-Execution-based Testing (10 points)

*Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).*