Software Implementation and Testing Document

For

Group 16

Version 1.0

Authors:
Asher Adams
Robert Romero
Kevin Sturge
Valeria Torres

1. Programming Languages (5 points)

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).

For this project, we are using GDScript, which is the language supported by the Godot engine. GDScript is useful in making the visual game components and logic. C# is also used in the dungeon generator part of our project.

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

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).

Platforms: Godot, Github, Trello

Technologies: Git, .net, Asprite

We program and test everything in Godot and do version control and collaboration in Github and Trello. Trello has our kanban board and information on each card. Aseprite has been used to create the player sprites and dungeon tiles.

3. Execution-based Functional Testing (10 points)

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

Functional testing was performed to the best of our ability by running the program in isolated scenes as well as through the overall scene depending on which element was being tested. We playtested to ensure implementations are working correctly with no chance for softlocks, bugs, etc. The game loop is in the process of being implemented and as we go, we are testing it thoroughly by running through the game.

4. 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).*

Debug statements and analysis of our game's performance as well as user experience were how we performed non-functional testing. As well as print statements in the code to make sure certain parts were working.

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

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

At various points during development, we have done code reviews where we go over all the recent changes to the project and make sure that the new systems work with the existing codebase. This is important when merging pushes to make sure that the code is compatible even without running the code, and it avoids code silos.