Testing and DevOps Plan

# Existing Functionality

Planet Ball is a game built by Eryl Kenner and Kyle Brainard for the Reno 2018 Hackathon. It is written in C# using the Unity 2017.4.14f1 game engine. To play the game you need two players using either controllers or a keyboard, but testing can be done with a single person.

# Testing Plan

Since Unity is a developed product, I will not be testing that aspect of the game. However, I will be writing code to test the other scripts within the game.

### Unit Tests

This will be done using Unity’s built-in Unity Test Framework, which utilizes the NUnit framework. 75% test coverage will be based on having tests for 75% of the functions in the game.

These are the aspects of the game I will test:

* Scoring
* Human input
* Player collision
* Player orbit physics
* Ability timing
* Ability usage
* Particle effects

### Integration Tests

These are the integration tests I will add:

* Human input causes player movement
* Human input causes ability usage
* Scoring causes particle effects

# Automation Plan

### Centralizing the Code

All code for the project is stored on GitHub <https://github.com/Eryl2000/Planet-Ball>.

### Continuous Integration

Tests will be run automatically in a docker container using Travis CI when pushing changes. I looked at several ways to do continuous integration with Unity and found a GitHub project which explains how to do it fairly simply using Travis CI, which is why I chose to do it that way <https://github.com/GabLeRoux/unity3d-ci-example>.

### Automated Build

The project will be automatically built as a standalone windows x86 binary using Travis CI. The reason for this is because it will use the same tools from continuous integration (Travis CI, Docker, GitHub), which will streamline the project. I found a useful article on how to do this <https://medium.com/@carlos_c/how-to-build-and-release-a-standalone-unity-app-using-travis-ci-docker-and-github-7798207bb865>.

Note: Since the game is meant as a fun local coop game, deploying it to a server does not make any sense.