Game Project DevOps Plan

Game Title:

Team six:

1. Introduction

This project is about creating a video game using Godot Engine while also applying DevOps practices. That means we’ll use tools that help us work more effectively as a team, test our game quickly, and ensure that updates are seamless.

We’ll use GitHub to manage our code and art files. Every time we make a change, GitHub Actions will help us build the game automatically and run tests. Discord will send us alerts if something breaks. We’ll test the game both by hand and with a tool called GDUnit3.

The goal is to have a fun and stable game that we can keep improving easily — just like real game studios do, but using tools and methods that make sense for student teams.

2. What the Game Needs

The game should feature engaging gameplay, including player movement and a basic combat system. Players should be able to access menus and view useful information on the screen with a clear and simple interface.

Behind the scenes, we want our DevOps system to help us every step of the way. When we push changes to GitHub, it should automatically test the code and create builds for Windows and Linux. If something goes wrong, we’ll get bug reports through GitHub Issues or Trello.

For communication, our team will use Discord to talk in real time and Trello to organize tasks. The game itself should remain small and function well, even on basic computers with built-in graphics.

3. How Everything is Built

The game will be organized into scenes in Godot: one for the player, one for the world, and one for the menus. We’ll write the code in GDScript, utilizing object-oriented principles to maintain a clean and manageable structure.

On the DevOps side, GitHub will be our main workspace. We’ll create different branches for new features, and when we finish something, we’ll merge it into the main version. GitHub Actions will automatically build the game and run tests. If a test fails, our Discord will let us know. Trello will help us track progress, with cards linked to GitHub Issues.

4. Timeline (6 Weeks)

During the first 3 weeks, we’ll create a simple version of the game that you can play. At the same time, we’ll set up GitHub Actions. In the second month, we’ll improve the game’s menus and add automated testing with GDUnit3. In the last 3 weeks, we’ll focus on playtesting and making the builds better.

After the game is released, we plan to fix any issues players find and maybe even set up an automatic way to publish updates, like on itch.io.

5. Our DevOps Workflow (SDLC)

We’ll start by picking the right tools — GitHub for code, Trello for task planning, and Discord for team chats. Then we’ll decide what features matter most and design how the game and DevOps tools will connect.

We’ll build the game in small pieces every week. Each piece will go through testing using GDUnit3 and hands-on playtesting. Once it’s ready, GitHub will help us build and deploy it. After release, we’ll continue to monitor logs and player feedback to identify and address bugs, and further improve the game.

6. Tools, Setup, and Team

We’ll be using Godot 4.x on regular PCs, with approximately 1 GB of space allocated for builds. The game’s main language will be GDScript, but we might use C# for some parts.

.