Good Mood Games - Junior-Mid Level Game Developer Test Case

Project Overview

Position: Junior-Mid Level Game Developer
Theme: Combat System with Training Dummy (Inspired by Hades)

Technical Requirements

Development Environment

- **Unity Version:** 6 or higher
- **Render Pipeline:** HDRP (High Definition Render Pipeline)
- **Programming Language:** C#
- **Input System:** New Input System
- **Camera System:** Cinemachine

Submission Requirements

- Complete Unity project files
- Playable Windows build
- Project shared via GitHub repository

Project Summary

This test case requires the development of a third-person action game prototype featuring a character control system, dynamic camera management, and simple combat mechanics. You will create a player character that moves fluidly through a 3D environment using keyboard controls with free-look camera functionality, implement a lock-on camera system for combat encounters, and design a basic combo-based sword combat system that allows players to chain attacks for increasing damage. The combat system will be demonstrated through interaction with a training dummy that features a health system, visual damage feedback, hit reactions, and automatic respawning. The goal is to showcase your ability to integrate multiple game systems - character movement, camera control, combat mechanics, UI elements, and object interaction - into a cohesive and polished gameplay experience that demonstrates both technical proficiency and game feel sensibilities.

Core Gameplay Features

1. Third Person Character Controller

Movement System:

- **WASD Controls:** Forward movement
- **WASD Controls On Camera Lock:** strafing movement
- **Mouse Look:** Camera control and character rotation
- **Movement Base:** Unity CharacterController component
- **Character Rotation:** Align with movement direction
- **Reference:** The Witcher 3 character movement

Camera System:

- **Free Look Camera:** Default third-person perspective
- **Lock-On System:** Tab key to lock camera onto enemy
- **Cinemachine Integration:** Smooth camera transitions and controls

2. Combat System

Attack Mechanics:

- **Basic Attack:** Left mouse button to perform sword attacks
- **Combo System:** Chain attacks by pressing attack button just before current attack ends
- **Final Combo Hit:** Significantly higher damage on last attack in sequence

Animation Requirements:

- Sword attack animations for each combo stage
- Smooth transitions between combo attacks

3. Training Dummy System

Dummy Behavior:

- **Static Object:** Dummy remains stationary
- **Hit Reactions:** Visual feedback when struck
- **Health System:** Takes multiple hits before death
- **Death Animation:** Plays death sequence
- **Auto Respawn:** Automatically respawns 5 seconds after death

Health & UI:

- **Health Bar:** On-screen UI displaying dummy's current health
- **Damage Feedback:** Visual indicators showing damage dealt
- **Real-time Updates:** Health bar updates when damaged

Bonus Features

Enhanced Game Feel

- **Camera Shake:** Impact feedback on successful hits
- **Visual Effects:** Blood splatter, impact particles, weapon trails
- **Audio:** Sound effects for attacks, hits, and dummy destruction
- **Additional Animations:** Varied attack animations for visual diversity
- Rest is left to your imagination

Assets

Use third-party assets for all requirements as needed.

Submission Instructions

Repository Setup

1. Create a public GitHub repository
2. Include the Unity project
3. Provide clear README with setup instructions
4. Include playable Windows build

Email Submission

- Send repository link to designated email address
- Include brief description of implemented features
- Mention any challenges faced and solutions implemented