Project Overview

This document explains the main systems of the prototype developed over the last 48 hours. The project focuses mainly on player movement, world interaction, combat, and an inventory system.

Player Movement and Camera

Player movement is intentionally simple and uses Unity's CharacterController. The player can move, rotate, and attack. The camera is implemented using Cinemachine, which allows smooth mouse rotation and collision avoidance.

An additional AnimationHandler script is used to manage animation events. This script is shared by both the player and enemies and is responsible for triggering gameplay and audio events during animations.

Inventory and Equipment

The inventory system uses a decoupled architecture. InventoryManager handles all inventory data and logic, while InventoryUI is responsible for the visual representation. InventorySlotUI manages drag-and-drop interactions.

The system supports stackable items, item swapping, and equipping items into special slots such as weapons, armor, and consumables. A CharacterVisualManager is used to decide whether an item should be instantiated (for example, weapons) or simply shown or hidden on the character model.

Combat and Health

The combat system is based on an interface called IDamageable, which allows any object (player, enemies, or environment objects) to receive damage using the same logic. Weapons use trigger colliders to detect hits and ignore the owner to prevent self-damage.

To improve game feel, a short hit stop effect and screen shake were implemented using Cinemachine Impulse.

Data Persistence

Game progress is saved using JSON serialization. The inventory state is converted into a string and stored using PlayerPrefs. When the game starts, items are reconstructed using Resources.Load, ensuring that inventory and equipment persist between sessions.

Conclusion

This prototype demonstrates a clean and scalable foundation. The use of interfaces and decoupled systems makes it easy to extend functionality and add new features without affecting existing code.