Project Overview, Development Process, and Self-Assessment

The project began with a brainstorming phase focused on defining a scope that could be realistically delivered within the evaluation timeframe. I initially explored multiple concepts and refined them into a roguelike arena game featuring fixed enemy waves and a preparation area outside the arena where the player manages equipment before each run. This structure was chosen to provide replayability through systemic variation rather than content volume, allowing multiple unique playthroughs without requiring constant addition of new systems.

During development, I reassessed the original scope and made deliberate cuts to ensure stability and core feature completeness. Systems such as enchanted items, wave modifiers, and a final boss were removed after identifying that they would compromise delivery within the given deadline. This decision allowed me to focus on building a solid and extensible foundation.

Given the limited timeframe, some systems were designed with extensibility and clean code in mind rather than minimal implementation, which impacted delivery time but resulted in a stronger foundation.

The entity, stat, and item systems are deeply interconnected and represent the most time-consuming part of the project. In hindsight, this area would have benefited from more upfront planning, as it contributed to delays later in development. However, the resulting architecture allows players, enemies, NPCs, and interactive objects to share a common base, simplifying interactions and future expansion. Items can directly modify entity stats, including newly introduced or entity-specific attributes, without requiring structural changes.

The combat system, integrated with the item system, supports multiple weapon types with distinct behaviors. By configuring a ComboData object, it is possible to define unique attack patterns that significantly alter gameplay feel. Combined with tiered stats, this enables clear progression and balance control.

The inventory system is based on Scriptable Objects to ensure organization, easy editing, and scalability. Newly created items are automatically registered in the item database and fully supported by the save system without additional configuration. The UI layer reflects inventory data consistently and supports drag-and-drop interactions and contextual tooltips.

To streamline development, I reused previously developed utility systems such as DropSystem, InteractionHandler, SceneLoader, and DataCarrier. Third-party assets from Synty Studios and animations from Mixamo were also utilized.

In conclusion, while the project could have been more refined with stricter scope control, the resulting systems are robust, reusable, and well-structured. With additional time, this foundation could be expanded into a polished vertical slice with minimal architectural changes.

Willian Jorge Gonçalves da Silva