🎮 Game Engine Course – Self-Evaluation

**Student Name:**  Yifei (Jeff) Ma   
**Project Title:** 2DGameEngine   
**Date:** August 13, 2025

1. **Collision Mechanics**

I, Jeff Ma, **deserve a mark of 100%** for *collision mechanics*.  
I demonstrate competency in this area in the game I made by:

I believe I have earned 100% for the collision mechanics implementation based on comprehensive functionality, excellent code quality, and complete fulfillment of all core requirements. My AABB collision detection system demonstrates advanced understanding through its sophisticated layer-based filtering mechanism, which allows precise control over which entities can interact—essential for complex game scenarios like distinguishing between player-enemy, enemy-enemy, and projectile interactions. The event-driven collision response architecture showcases professional-grade design patterns, enabling clean separation of concerns where the collision system detects intersections while game-specific components handle responses through a robust publish-subscribe pattern. This architecture is evidenced in the ZombieSurvivor game where bullets correctly damage enemies, pickups trigger collection events, and solid objects block movement appropriately.

The implementation exceeds basic requirements through performance optimizations including spatial partitioning for efficient broad-phase collision detection, minimizing unnecessary checks in dense game scenarios. The differentiation between trigger and solid collisions adds gameplay depth, allowing for both physical barriers and interaction zones without code duplication. Code quality is demonstrated through consistent use of modern C++ practices, clear component-based design following ECS principles, and seamless integration with the engine's physics and event systems. The ZombieSurvivor game serves as a practical demonstration where all collision types function correctly: enemies collide with walls, bullets trigger damage on impact, power-ups activate on overlap, and the player experiences proper physical responses. This complete, optimized, and architecturally sound implementation justifies the full 100% grade.

2. **Animation**

I deserve a mark of **100%** for *animation*.  
I demonstrate competency in this area in the game I made by:

I believe I have earned 100% for the animation system implementation based on its sophisticated multi-component architecture, technical excellence, and comprehensive demonstration of animation concepts within

the ECS framework. My animation system showcases advanced understanding through its multi-layered approach: a robust sprite state system that manages complex animation transitions, dynamic sprite switching that allows runtime changes based on game events, and precise frame-based timing that ensures smooth, consistent playback regardless of framerate variations. The architecture elegantly separates concerns with dedicated components for animation data (AnimationComponent), sprite rendering (SpriteComponent), and state management, all orchestrated by specialized systems that respect ECS principles. This design enables complex animation scenarios like directional movement sprites, attack animations with precise timing windows, and seamless transitions between idle, moving, and action states—all demonstrated effectively in the ZombieSurvivor game.

The technical execution demonstrates professional-grade quality through features like animation event callbacks for synchronizing game logic with specific frames, support for both looping and one-shot animations, and efficient sprite atlas management that minimizes texture switching overhead. The ZombieSurvivor implementation serves as a comprehensive showcase where player characters smoothly transition between movement directions, enemies display death animations that trigger appropriate game events, weapon effects animate with proper timing, and UI elements utilize animated sprites for visual feedback. The system's flexibility is proven through its ability to handle diverse animation requirements—from simple two-frame idle animations to complex multi-directional sprite sheets—all while maintaining clean, maintainable code that follows modern C++ best practices. This complete implementation, which seamlessly

integrates with the engine's rendering pipeline and event system while providing rich gameplay animations, fully justifies the 100% grade.

3. **Physics**

I choose to be evaluated on: Physics

I deserve a mark of **95%** for this area.  
I demonstrate competency by:

I believe I have earned between 90-95% for the physics system implementation based on strong technical execution and comprehensive feature coverage, with minor areas that could be enhanced. My physics implementation demonstrates solid competency through key systems: exponential friction damping that provides smooth, realistic deceleration for game entities, a velocity constraint system that effectively prevents unrealistic speeds while maintaining gameplay responsiveness, and successful integration with the collision response system for proper physical interactions. The particle physics support adds visual richness through effects like debris and projectiles that follow believable trajectories, while boundary constraint physics reliably keeps entities within the game world through position clamping and velocity adjustments. These systems work cohesively in ZombieSurvivor, where player movement feels responsive, projectiles behave predictably, and enemies navigate the environment with appropriate physics-based movement patterns.

While the implementation is functionally complete and demonstrates all required concepts effectively, I acknowledge areas where additional refinement could elevate it to 100%. The physics simulation could benefit from more advanced integration methods (currently using basic Euler integration which can accumulate errors), the particle system could support more complex behaviors like particle-to-particle collisions, and the friction model, while functional, could be enhanced with separate static and kinetic friction coefficients for more nuanced movement. Additionally, while the system handles standard gameplay scenarios well, extreme edge cases like very high velocities or multiple stacked constraints occasionally show minor instabilities. Despite these areas for potential improvement, the physics system successfully delivers smooth, stable gameplay with all core requirements fully implemented and integrated, justifying a grade in the 90-95% range based on its strong foundation and practical demonstration in the game.

Reviewer Acknowledgment

I had this self-evaluation reviewed by: **Alex Hortua**