SkateKing - System Explanation and Personal Assessment

System Explanation

The Skateboarding Simulator Game is developed using Unreal Engine 5.3 and C++. The primary focus is on creating realistic physics-based movement for a character on a skateboard, incorporating actions like speeding up, slowing down, jumping, and earning points for jumping over obstacles.

The core components of the system include:

1. **Movement System**: Utilizes physics-based calculations to simulate realistic skateboard movement. Acceleration, deceleration, and friction are applied to control the character's speed.
2. **Jump Mechanics**: Includes a Mixamo jump animation, smoothly integrated using an Animation Blueprint. The jump logic ensures the character can only jump again after landing.
3. **Point System**: Points are awarded based on the height of obstacles the character jumps over. This is dynamically handled using raycasting and sphere tracing during the jump.
4. **Collision Handling**: Ensures smooth movement and realistic collision responses using the SafeMoveUpdatedComponent method in the UCharacterMovementComponent.
5. **User Interface**: Displays points earned by the player, providing immediate feedback on their performance.

Thought Process

During the interview, my primary goal was to create a robust and realistic skateboarding experience. Initially, I experimented with the Character Movement Component and the Vehicle Movement System provided by Unreal Engine. However, skateboarding mechanics are unique and require precise control with realistic physics interactions, which these components couldn't fully provide.

Therefore, I decided to develop a custom movement system from scratch. This involved integrating physics, collision detection, and ensuring custom control tailored to skateboarding dynamics. The challenges included integrating the jump animation, handling continuous jumping issues, and ensuring smooth collision detection.

I approached these challenges with a modular strategy: setting up movement, integrating animations, implementing collision detection, and refining the point system and UI. This iterative approach allowed me to identify and resolve issues effectively, ensuring a polished and responsive gameplay experience.

Personal Assessment

I believe my performance was strong. I effectively utilized Unreal Engine's features to create a realistic skateboarding simulator. The integration of a custom physics-based movement system, animation blending, and point system demonstrates a good understanding of game development principles.

Time Spent:

- Movement System: 4 hours
- Jump Mechanics: 2 hours
- Point System: 2 hours
- Collision Handling: 5 hour

- UI Integration: 2 hour
- Debugging and Refinement: 2 hours
- Documentation: 1 hour

Total Time Invested: 18 hours

Overall, I am pleased with the outcome and confident in my ability to tackle similar tasks in the future.