USkateMovementComponent: A custom character movement component for arcade-style skate physics, replacing the default walking mechanics. It ensures the character always moves in their facing direction, eliminating sliding. It manages custom acceleration, braking, and a charged jump mechanic.

The project includes a flexible scoring system for jumping over obstacles. These obstacles are defined by a custom component and identified using GameplayTags. A UDataAsset maps these tags to score values, allowing for easy modification by designers. Upon landing, a BoxTrace along the jump's trajectory detects all cleared obstacles. Duplicate points are prevented by a TSet that tracks objects scored within a single combo, combined with a time-based cooldown to ignore rapid, unintentional bounces.

The architecture follows best practices for data and UI communication. A custom ASkatePlayerController is responsible for creating the UI. The player's score is stored in a custom ASkatePlayerState, which uses a multicast delegate (OnScoreUpdated) to broadcast changes. A C++ UPlayerScoreWidget subscribes directly to this delegate for efficient, decoupled updates.

Finally, at runtime, a separate Skateboard Blueprint actor is spawned and attached to a socket on the character’s skeletal mesh, keeping the visual representation modular.

It took me some time to set up everything initially and to choose the best scene.
For focus reasons, I ended up going with a low-poly scene.

The part that took me the most time was the logic behind the skateboard movement and managing the interaction between the CharacterMovementComponent and
the SkateMovementComponent.

The rest flowed more easily.

Thanks for the opportunity.