System Explanation

Skate Movement (2d)

The skate movement was created to implement a custom character movement component. The idea was to simulate skateboarding movement with friction, impulses generated by player input, and a gradual slowdown effect as the skateboard loses velocity over time.
I created a custom character and implemented the basic input system using the Enhanced Input System.
Additionally, I implemented a custom animation notify to make it more realistic when the animation applies an impulse.
This topic took the longest to implement because I had never worked on creating a custom movement before, so I first researched and studied the basics to understand how it works.

Point System (5h)

I created an actor called **S_Obstacle**, which contains triggers and delegates.
These have two important features:

1. Call the GameMode to notify that an obstacle was passed by the player.

2. Deactivate the object and hide it once the player passes it.

The GameMode holds references to all **S_Obstacles**. It manages adding points and notifying, through its delegate, that the score has changed. The widget and UI then receive this notification and update the displayed score accordingly.

HUD (2h)

I created two widgets: the **Win Panel** and the **HUD**.
I designed them and implemented their calls in the GameMode and PlayerController, respectively.
The HUD includes controls and the score display, showing both the total number of obstacles in the map and how many have been passed by the player.
For now, the Win Panel is simply a panel that informs the player they have completed all obstacles, offering the options to try again or exit the game.

Personal Assessment

I think I did a good job implementing a system that I had never worked on before.
I feel I could improve my implementation speed if I refresh my Unreal Engine knowledge, since I stopped working with it about five months ago to focus on Unity.
However, a bit of practice was useful.
Due to time constraints, I know I can still improve the system — for example, by rotating the character according to the surface, or making the component more modular and generic.