***Technical - Explaining the System:***

The way that it is commonly used for physics controls is to code a custom character movement mode (either from scratch or expanding Unreal's Character Movement Component) and use physics to drive the movement. But given the short number of hours that was available I decided to implement a basic physics driven controls by using a sphere mesh that is used to drive the physics movement, and then set the character location to it each frame. It is not the ideal way to do this, but it is a way that works and could be finished in time. Also added the possibility to enter and exit the skate at will, returning to normal controls and movement.
Other descriptions of key aspects of the system:

- Acceleration and braking works through a force being applied to the ball.
- Jump works by applying an impulse to the ball.
- For the scoring, it just sums and stores the points earned by jumping obstacles.
- The detection of jumped obstacles is done by a simple box overlap, but with more time the ideal and safest way would be to detect obstacles in front of the player and then do some vector math to check if it was jumped over.

***Time Report:***
I received the task on Friday at 15:00pm (GMT -3), so given the instructions I had until Sunday 15:00pm (GMT -3) to deliver. I could not work on it on Friday, so basically all tasks were done on Saturday.

Core Skate Movement: **4:30h**
Jumping Over Obstacles: **2:30h**
Animation Setup: **1h**
Skate Arena Blockout: **1h**
Writing this document: **1h**
Total hours spent: **10h**

***Further Development:***

What I would do with a bit more time (maybe five days) :
System Structure:

- Would setup the skate logic in an actor component instead of directly in the character class

Visuals and Aesthetic:

- IK setup for the player legs and skate
- Landing animation when the character lands from a jump or height drop

Gameplay:

- Currently the forward physics force is applied constantly while the input is pressed, which does not look so great. I would instead use each foot "step" of the forward animation to add impulse on the player, which would look more natural.
- Ramps impulse and simple tricks detection (360, 720 turns): Detect when the player accelerates and leaves a ramp, manage the outwards speed to give the player more upwards speed than forward speed and be able to do 360 or 720 turns by holding right or left and landing on the ramp afterwards.