Agustin Ruscio Skate Club

SkateClub Development

Development Time (Approximately):

- Movement 2h
- Obstacles 1h
- Obstacle Feedback 3h
- GameLoop 2h
- Environment 1h
- UI 3h

Movement:
The movement code is separated between the controller and the character. The Controller takes the input Action and binds it to the context, and with delegate calls the listener. The character as a Listener performs the movement logic. So in case of a Second player or different movement option, a new character should only subscribe to the controller delegates.

Obstacles:
A parent obstacle is developer, with box components to detect if the player overlaps. Then a blueprint derived class is create to be a model for the child blueprints of the class, so designer and developer do not have to create from the C++ class and in case of needing a change for all obstacles, just changing the parent every single obstacle would change.

Obstacle Jumped FeedBack:
One of the most important part in my opinion. UI Pop ups, sound fx and cameras shake. In case of jumping an obstacle the character make SFX, and display cheers ui, the controller make a camera shake and display the obstacle info and the player state counts the score

GameLoop:
Every game needs a menu, and an end game. A timer is running. And when the time is over, the end game is displayed with score information.

Code structure:

The game system tries to be as generic as possible to avoid class dependencies. Using the Unreal Engine classes to take care of each own task. Separating the task in order to have a clear code and to avoid enormous code blocks.

Personal assessment:
Due to some personal challenges, I wasn't able to use the entire available time effectively. Environment design is not my strongest skill, so the result could have been better. I consider myself more of a programmer, with a stronger inclination toward UI development.
I structured the code to be as designer-friendly as possible, ensuring they can work without diving into the code. All developed features are clearly exposed in the editor for easy access.
The most polished aspects of the game were feedback and gameplay, along with the overall code structure. I focused heavily on elements such as camera FOV changes, sound effects, and UI pop-ups to enhance player experience.