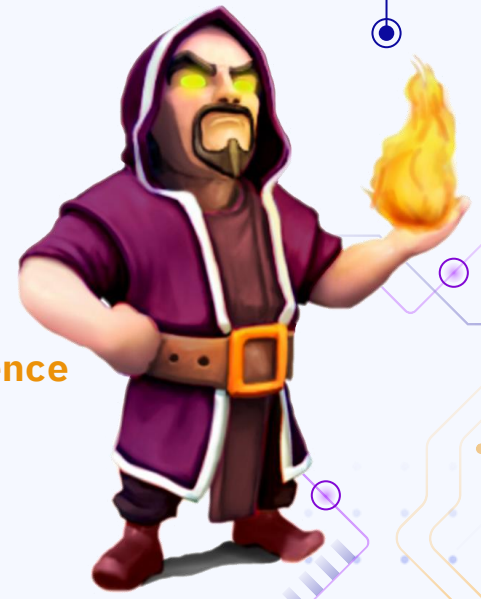
Multiplayer Networked Game

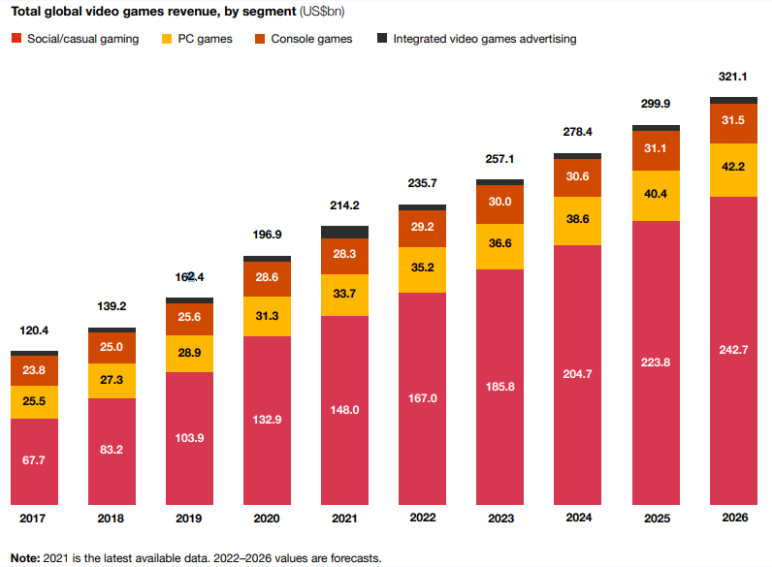
Casting the illusion of sharing the same expeirence

Presented by Kristupas Stalmokas
Supervised by Xiang Li



Introduction & Motivation

Online multiplayer games play a **major part** in the modern gaming industry, leading the way for player engagement, social interaction and **commercial success**.

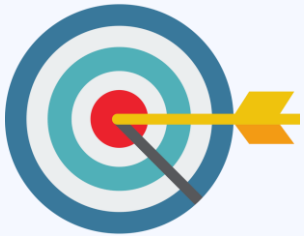


Despite cosmic evolution in the field, the problem of **ensuring a seamless, synchronized and fair experience** for all players remains a substantial technical challenge.

Project Aim & Objectives

AIM

Design and develop a **real-time, physics-based** multiplayer networked game that offers a **seamless, synchronized and fair** online experience.



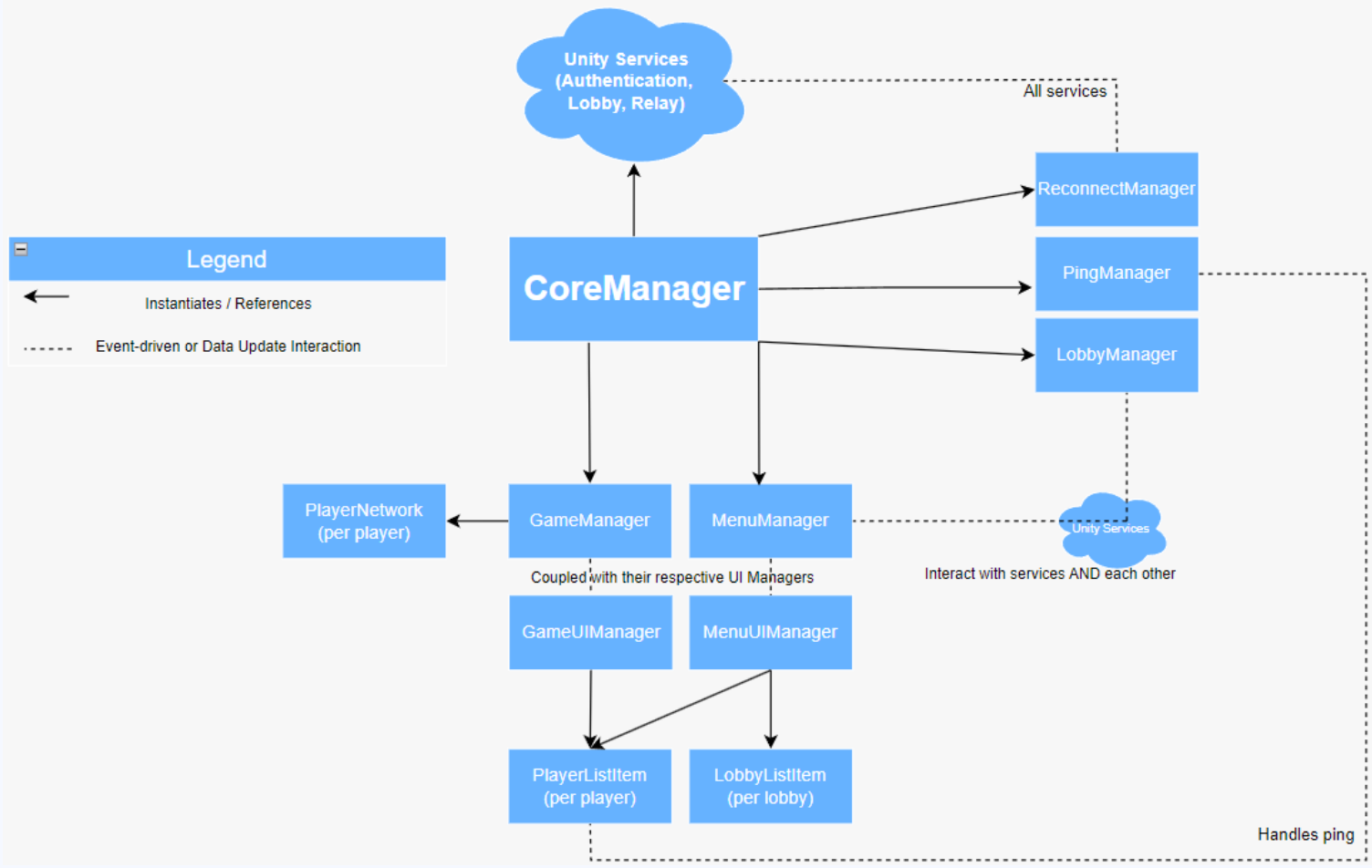
OBJECTIVES

- Allow players to find each other online and agree to start a game.
- Efficiently **share the game state** and restrict cheating.
- Ensure the **network is resilient** with unreliable or intermittent connections.
- Ensure **real-time synchronization** between the player's views of the game world.

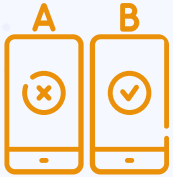
System Architecture

- Modular, Manager-based Design:
  - Each core system managed by their own dedicated singleton class
- Client-server Topology:
  - Server-authoritative model
  - UDP underlying protocol with rUDP on top for data that cannot risk being lost
- Seamless Unity Service Integration:
  - Authentication for player identification
  - Lobby and Relay for matchmaking and secure host-client connectivity
- Efficient State Synchronization:
  - NetworkVariables, RPCs and custom serialization for bandwidth optimization
  - Network Techniques discussed later

Modular Manager Design



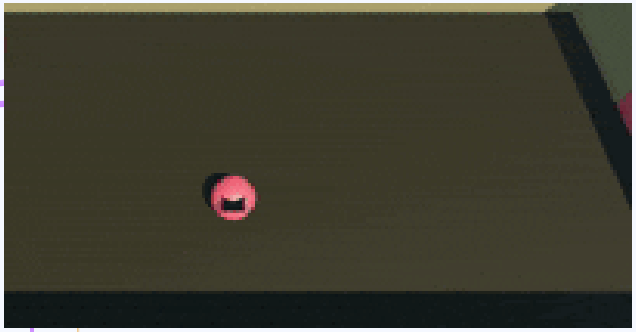
Networking Techniques



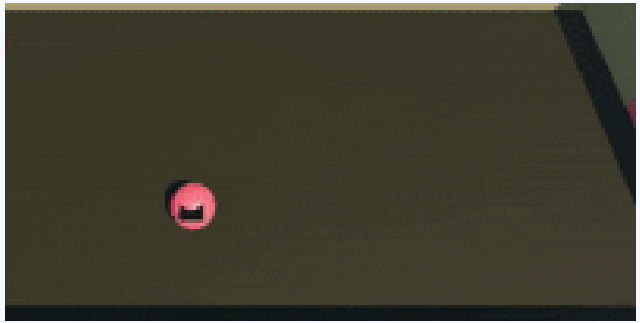
Client Prediction for player's movements and actions to be applied instantly on their own state to achieve responsiveness before the server responds.

Server Reconciliation for consistency after these predictions by correcting the differences between the client's predicted state and the server's authoritative state.

No Techniques: major input lag and stutter



Prediction Only: Minor input lag, major stutter



Both Techniques: Minor input lag, minor stutter

Security, Resilience & Reconnection

Server-authoratitive Logic
Critical game state changes validated only on server.

Dynamic Network Resilience

Each client's ping calculated and displayed.
Adaptive network parameters – change resolution and prediction / reconciliation parameters suitable for ping.

Reconnection & State Recovery

Players can reconnect and resume gameplay.

Privacy & Anti-cheat
Players identified via authentication and server-side validation.

Validation & Results

- Synchronization and playability maintained for clients up to 100ms ping and moderate packet loss.
- Smooth gameplay under suboptimal conditions through prediction and reconciliation.
- Reconnection successful to resume on-going session.
- Relay servers, dynamic network technique adaptability.

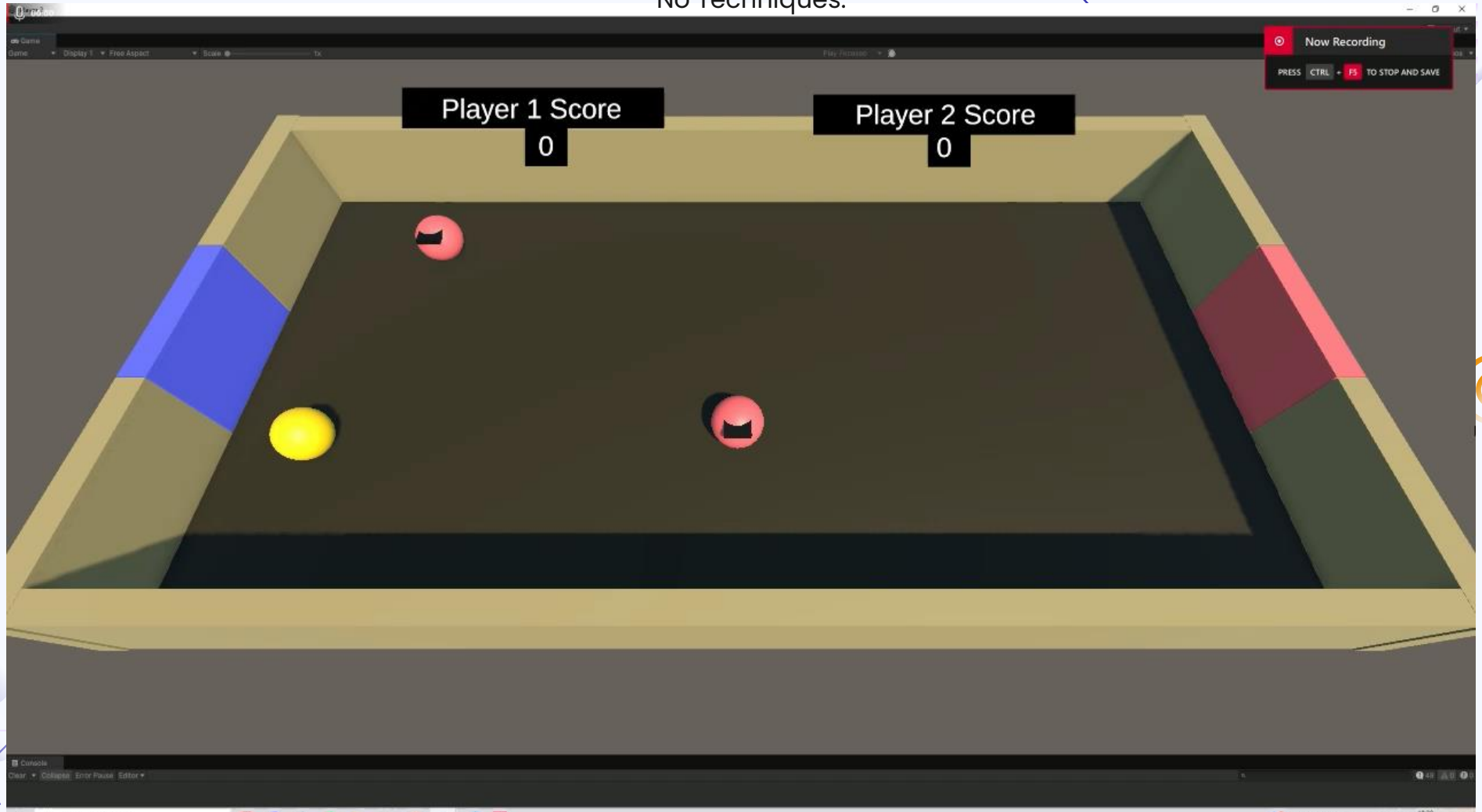
Conclusions

- A modular, manager-based architecture in the Unity engine can output a secure, fair and resilient multiplayer networked game.
- Networking techniques significantly enhance player experience under realistic network conditions.
- A practical and scalable foundation for future development.

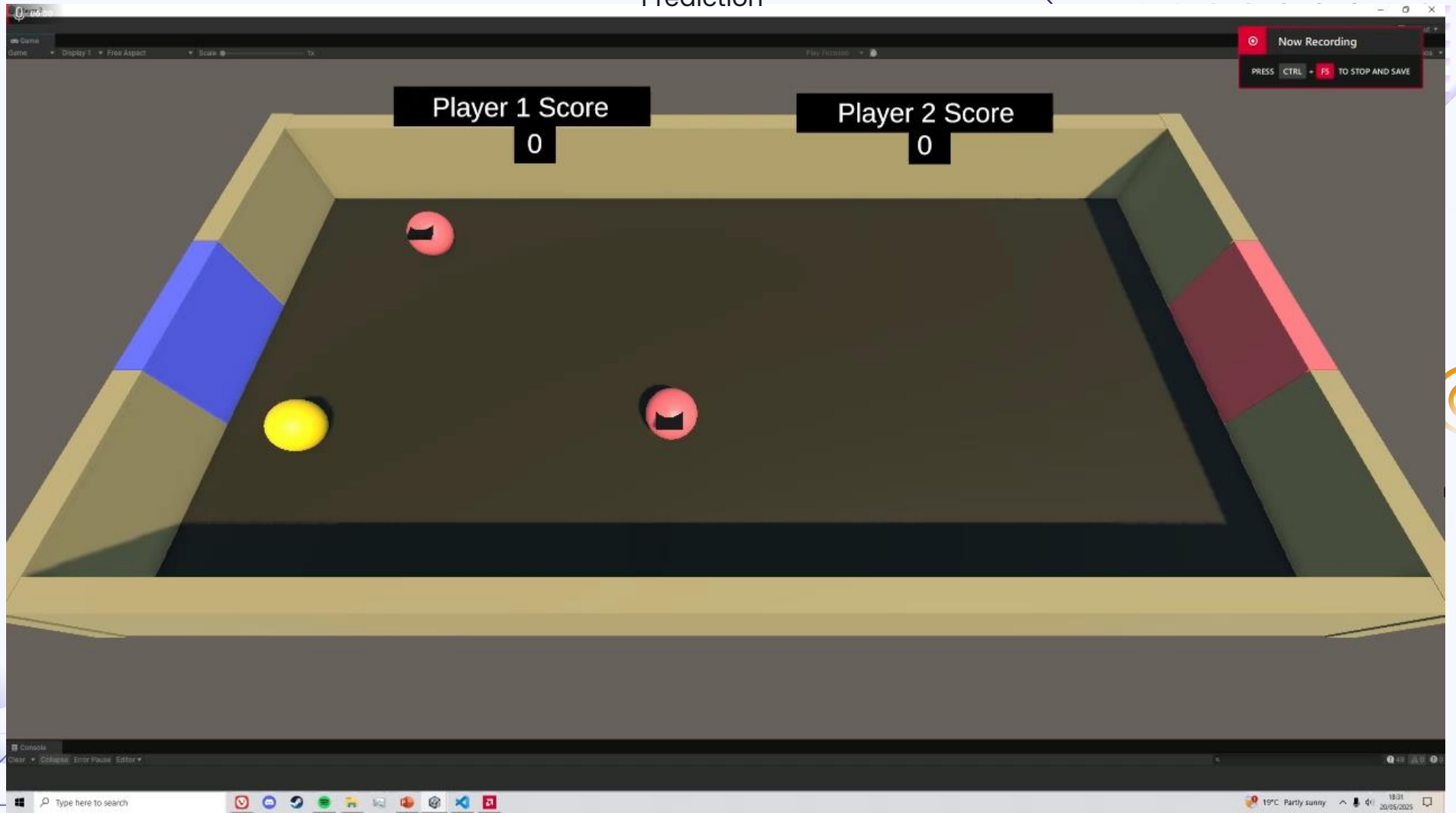
Thanks!

Any questions before demonstration?

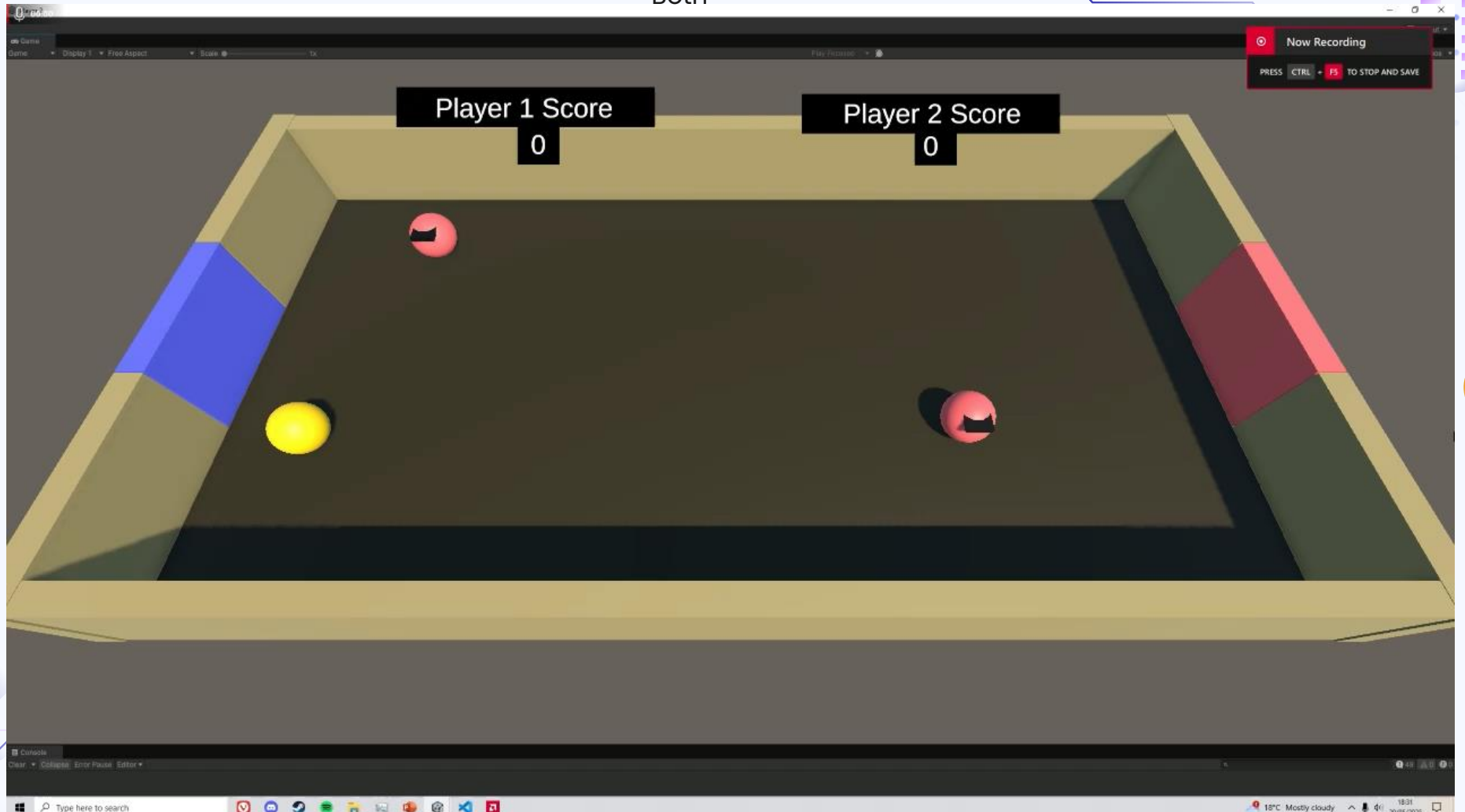
No Techniques:



Prediction


Now Recording
PRESS CTRL + F5 TO STOP AND SAVE
Player 1 Score
0
Player 2 Score
0

Both


Now Recording
PRESS CTRL + F5 TO STOP AND SAVE
Player 1 Score
0
Player 2 Score
0