**LasMeta Case Study: Dealer Interaction Prototype (Unity + Photon Fusion)**

**Context**: We aim to create a more immersive and realistic poker table experience in LasMeta. One important aspect of that experience is the presence and interaction of the dealer. In this case study, we’d like to see how you would bring this character to life in a lightweight prototype that integrates Photon Fusion for multiplayer functionality.

**Objective**: Using Unity and Photon Fusion, build a simple interactive multiplayer scene where the dealer responds to player actions through animations, audio, and UI. This will help us evaluate your skills in 3D scene composition, input systems, networking, UI interaction, and performance-conscious implementation.

**Tasks**:

**1. 3D Scene Setup**

* Create a basic poker table with a seated dealer character (can be a placeholder model). [Downloaded optimized model for poker take from sketchfab. Downloaded a dealer character from sketchfab too. Used mixamo, to add animation. Used a animation controller.]
* Add at least two player positions around the table (no models required; just coordinates or markers). [Used arrow marker indicating players position. That arrow is synced between players as that is NetworkObject. I would improve this. This is just a basic setup.]

**2. Photon Fusion Integration**

* Set up a multiplayer environment using Photon Fusion. [Used fusion 2. Used Fusion’s Connection GUI for quick setup and connection handle. Used Fusion Shared mode. OnPlayerJoined instantiated each player. Reeceived the event from NetworkEvent by subscribing to that. Although it was supposed to receive Player join callback from IPlayerJoined interface with SimulationBehaviour but That did not work. It was frustrating.]
* Each connected player should be able to trigger the “Deal Cards” action, synchronized across clients. [Each Player can trigger Deal Cards. But to make it safe I added some sense of state/turn control. On Clicking that, if the state is valid Cards were delt.]
* Ensure the dealer’s animations and sound effects are correctly networked to reflect state changes for all players. [On Clicking dealt after authority validation I changed dealers animation. For now POC I just changed to running anim. That is synced with rpc for now. Although this approach is kind of host authoritative, but I tried to mimic like server authoritative, like it should be in real game. With each card deal there is sound and that is also synced or other word invoked in each client.]

**3. Input System**

* Implement a “Deal Cards” action, triggered by either a keyboard key or a UI button. [Added UI button for Deal Card to get click from user. I used figma and created my own UI’s. Except for all the Cards. All other sprites are made by me using Figma. Also used optimized way of reusing already instantiated cards to reduce memory usages.]
* When triggered:
  + The dealer plays an animation (e.g., reaching toward the table). [The dealer plays a running Anim]
  + A sound effect plays (e.g., card dealing SFX). [Card Dealing Sound is added]
  + A basic visual effect simulates card distribution to the player spots (sprite or 3D object). [Card dealing is done by animating a 3D card object. I used DoTween to ensure its performance and control.]

**4. UI and Audio**

* Include a simple UI button labeled “Deal Cards.” [Added]
* Sync the animation and SFX when the button is pressed. [Added and synced]

**5. Roaming and Camera Control**

* Allow the player to freely move around the poker scene to inspect different angles. [Initially player joins with third person view. There is a toggle at the left that enabled forst person view and free roaming. Use Arrow Keys or WASD with mouse to move around.]
* Implement basic camera movement (orbit/zoom) around the table. [Camera control added. Player can zoom in/out and change different angle from the UI when in Third person view.]
* Add at least one adjustable light source and allow the user to change its direction or intensity during runtime (via simple UI or input). [One simple control for Spot light. Player can use a slider to change the intensity.]

**6. Bonus (Optional)**

* Add simple lighting, shadows, or shader effects to enhance immersion. [Shodows are there but didn’t add any fancy lights or shader. Used light baking for static objects to boost performance. For realtime lights used layer masks to restrict others getting affected by lights. Also that boost performance]
* Implement round-based dealing: Deal one card at a time to each player. [Round based dealing is added. One card at a time. Three in total for each. Also introduced turn/authority control. Used ReleaseStateAuthority and RequestStateAuthority to maintain that.]
* [Additionally I used Network properties to share information between clients. Although I wanted to use Change Detector and OnChange mechanism but there is not much documentation about these. It was another frustrating and challenging thing. At the end I had to sattle with RPC although I didn’t like that. There is a guide like Ui that I added. It also animates with DoTween and shared important game updates with the player. There is some validations added like button interaction are restricted based on turns or game state.]
* [There were a lot of other things I would do like lots of validations. Like if someone leaves between game, Someone joins between games and losts others. Finally I think I would go for a Fusion Server Mode approach to ensure the integrity.]
* [I used chat gpt and google for my research. Used chat gpt to give me a nice readme. But I provided all the information]

**Deliverables:**

* A Unity project folder (you may share via GitHub or Google Drive).
* A brief README or PDF explaining:
  + Your approach
  + Photon Fusion usage and synchronization logic
  + Tools and systems used
  + Any challenges encountered
* A short video (max 2 minutes) demonstrating the prototype in action.

**Evaluation Criteria:**

* Clean implementation of animation, input, UI, and networking systems.
* Logical integration and synchronization of components using Photon Fusion.
* Creativity and attention to user experience.
* Bonus features and performance awareness.

**Note**: This is not meant to be a production-ready feature. We want to see how you think, organize your scene, and structure interactions using Unity and Photon Fusion.

We're looking forward to seeing your creativity and technical insight!

You can send the links and documents related to your work to ali@lasmeta.io