

By Ohad Badhav, Elizaveta Sidorova, Snir Ben Nissim

Advisor: Moshe Sulamy

Game Development with Unity

Introduction:

**General Game Overview:**

* The Casino game offers an Innovative multiplayer experience where players can create their own gaming rooms or join existing ones. This platform facilitates collaborative gameplay, allowing users to come together and enjoy a variety of engaging casino games. Through seamless connectivity and innovative features, players can experience the thrill of the casino from the comfort of their own devices.

**Seamless Connectivity and Immersive Experience:**

* Our project stands out by swiftly connecting players with nearby friends through our location detection feature, fostering a seamless and immersive casino experience. This unique functionality removes the necessity of adding friends to a list; instead, players can effortlessly join the game together simply by being in close proximity. Furthermore, our game redefines the casino experience by allowing players to participate directly from their mobile phones, while the main "table" is projected onto a shared main screen. This innovative approach ensures that all players in the same room can fully immerse themselves in a genuine casino atmosphere, creating an unparalleled gaming experience.

**Enhanced Social Dimension:**

* While some multiplayer casino games exist, many lack the ability for players to join rooms with their nearby friends. In other words, while they may facilitate multiplayer interactions, they often overlook the opportunity for players to easily connect with friends in their vicinity. This limitation can detract from the social aspect of the gaming experience, as players may prefer to engage in casino games with familiar faces or acquaintances nearby. By incorporating a feature that allows players to join their nearby friends' rooms, our game enhances the social dimension of the casino experience, providing players with the opportunity to engage in a true casino atmosphere where the mobile phone serves as the player's hand and the main screen as the table. This addition distinguishes our game from others in the genre, providing a more immersive and enjoyable multiplayer experience.

**User Engagement and Room Management:**

* The focal point of the game revolves around creating and joining rooms. When creating a room, users are presented with a screen to select their preferred game, such as Poker or Blackjack, among others. Upon joining a room, users automatically join the nearby created room. Room creators have the ability to stream the game on a separate screen, displaying relevant features for all players, while individual mobile devices showcase each player's unique features.

**Game Variety and Interactive Features:**

* Our game offers casino games including Poker, Blackjack. Using our location detection feature, players can swiftly join games with their nearby friends, providing an authentic casino experience as they play alongside others at the same table/ same screen.

Our game utilizes nearby detection technology to create a unique and seamless gaming experience, enabling players to swiftly connect and enjoy the excitement of the casino with their nearby friends, without the need for complex friend lists or cumbersome invitations.

Our game introduces the mobile and main viewer feature to enhance the casino gaming experience, providing players with the unique opportunity to participate directly from their mobile devices while simultaneously enjoying a shared table view on the main screen. This innovative approach improves immersion, fostering a genuine casino ambiance where players can seamlessly interact and compete in real time.

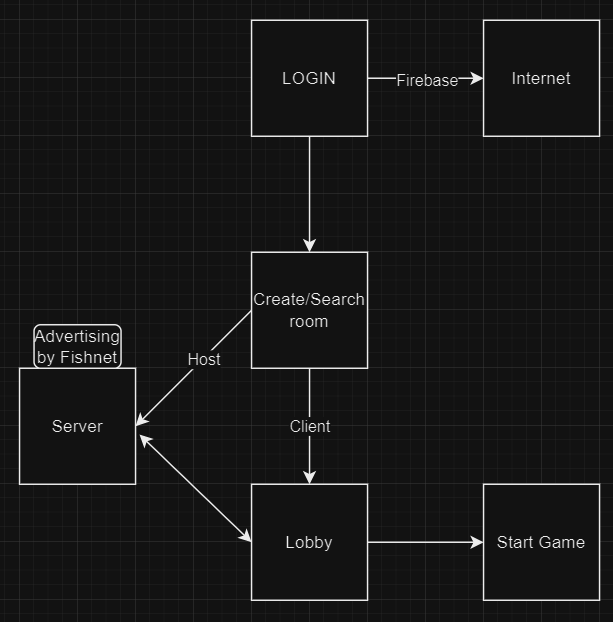
Top of Form

Architecture:

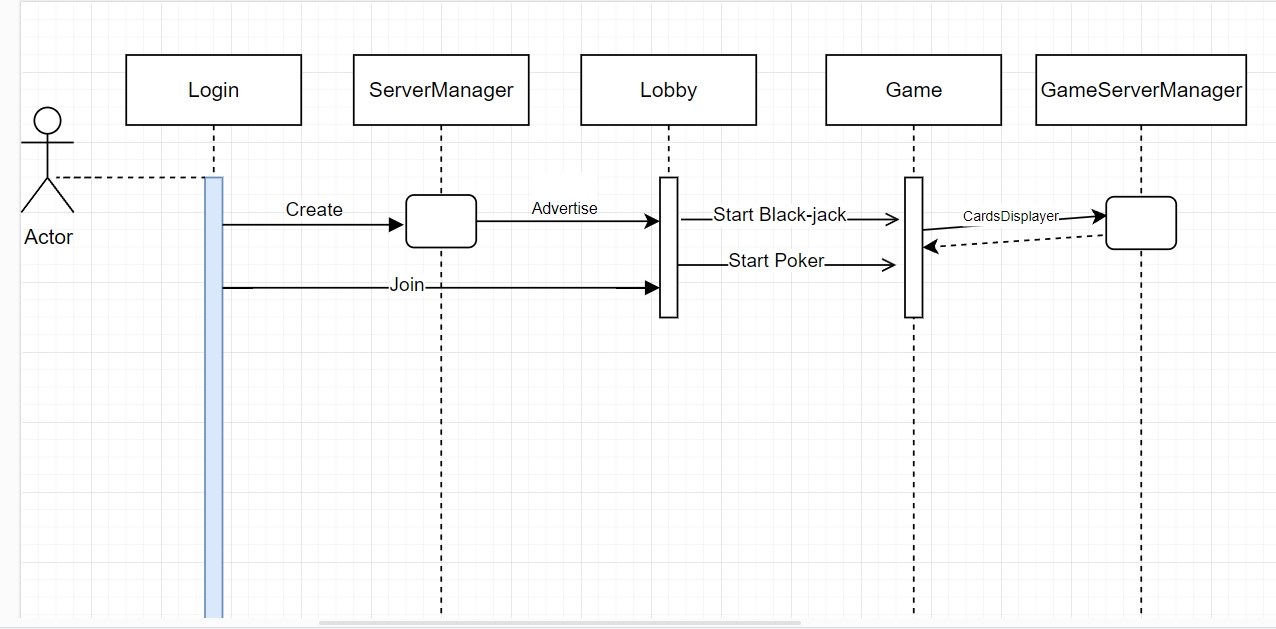
* We are Using the Unity engine, along with the C# scripts.

We use “Fishnet”- networking library for unity for communicating with nearby devices. We also use Firebase cloud service for authentication.

Our systems main components:



Sequence diagram of system interactions:



Code tests:

Login– In order to check the login logic, we signed in to Firebase’s authentication services, and in the firebase console we are able to see the signed in users, and the last login time. After logging into the game, we checked to see if the last login time was updated correctly.

That indicated the login works.

When a user signs up to the game, a new user appears in the firebase console in the authentication table. That helped us check our sign up implementation.

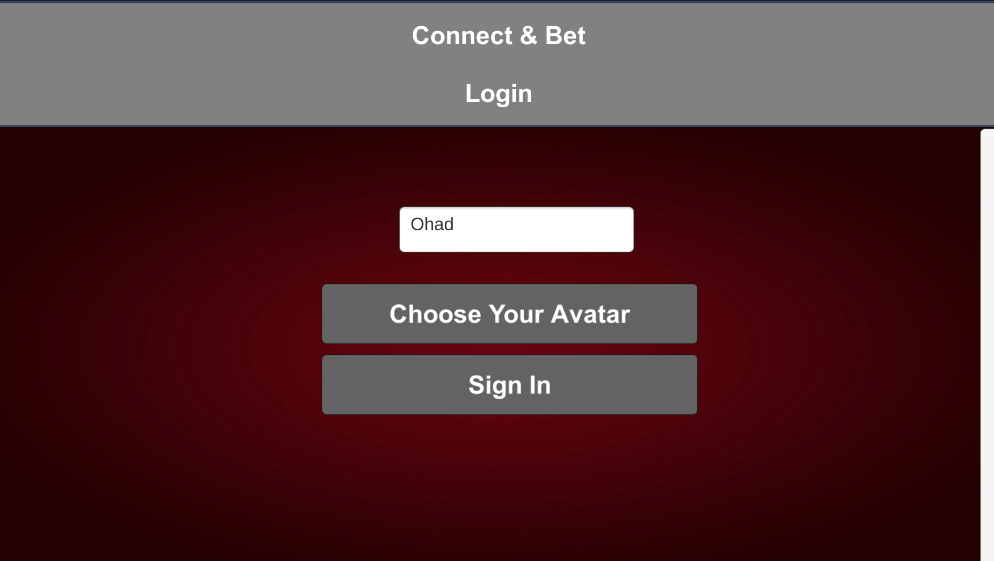
Our network detection was tested by creating an executable file of the game, running it from two different nearby devices.

Every client-server communication between the Host and the devices was checked by creating an executable file and running 2 or more instances.

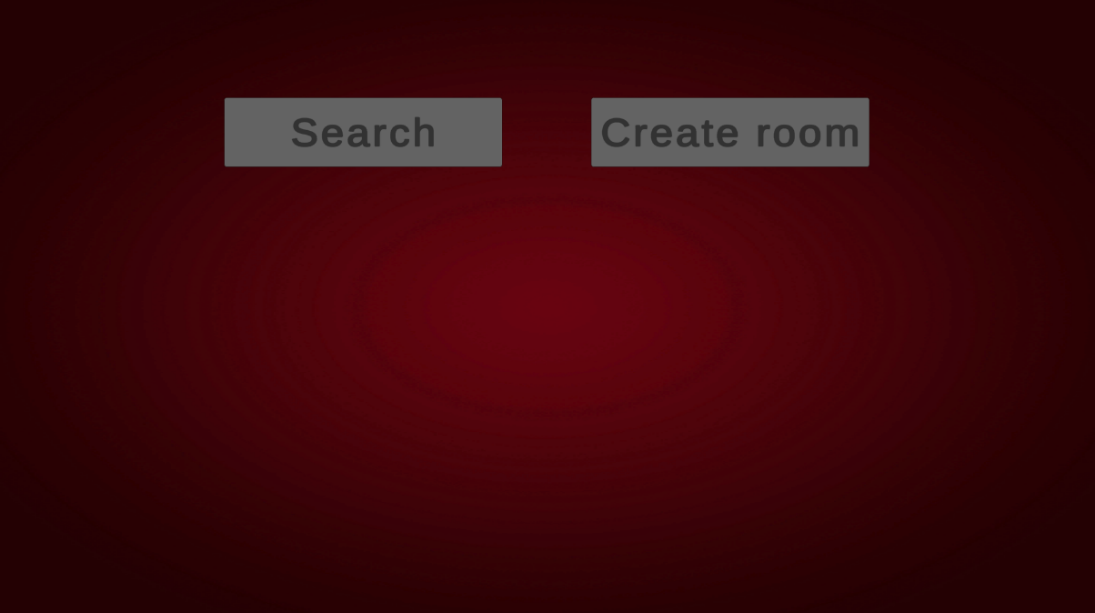
User guide:

At the beginning of the game, the user sees a login screen where they enter their name and (in the future) there will be an option to choose an avatar. From there we move to Room selection screen.

The login/sign-in screen:



Room selection screen - user is presented with two options. The first is to create a new room and wait for other players to join. The other option is to search for an existing room and join it.

User mode selection:

Host::

Once the host click the start button, all users move to the Blackjack room and the game begins.

Joining user:

Blackjack Room - shared screen



On the shared screen, the cards of all players, including the dealer, are displayed.

A player's view



In the mobile phone screen, each player sees the information relevant to them.

Hit: ask for another card

Check: skip your turn.

Double: double the bet.

Split: split the bet in case of cards duplication.

Our code contains 4 main classes:

**Login.cs**: A class attached to a script that runs on the beginning of the game, which provides the login and sign-up automatically implementation.

**ServerManager.cs** : Loads the network discovery script, which lets users connect to each other.

**MemberList.cs**: For every user joining the Host’s lobby, the memberList saves the LoggedUser.Username singleton instance as a string in a list, to be viewed in the Lobby room.

**GamerServerManager.cs**: The communication between every client and the server is relying on the syncObjects of the script.

How to run our game:

In order to run the game user needs to have the Connect&Bet.exe, UnityPlayer.dll, Connect&Bet\_Data folder, and MonoBleedingEdge folder.

User can download Connect&Bet.zip file, extract all files, and run Connect&Bet.exe.

Issues we faced:

An example of a problem we faced:

We looked for a way to search and connect user to the same game room, and interact between them without being connected directly to a main server and to the cloud.

The way we overcame this problem is by using the fishnet networking API.

One of the features that the fishnet networking API provide, is giving a User the Option to become a Host. The host then becomes a server that the other clients connect to, and interact with each other and with the cloud through him.

The fishnet networking API provides many other features that manage the game objects, and synchronize between them.

The known issues in our project:

One of issues we have in our project, is providing the option to play the game from a mobile device, which will be used as a “controller” for each player.

One of the ideas we have to improve our project, is creating other casino games that provide an interaction between different users, such as “Horse racing”, “Dice table”, and even sport games such as Soccer, basketball and other multiplayer games, which the players will be able to bet on the score based on fabricated data about un-real teams and leagues, and watch simulation of the game’s main events.

Other idea we have, is the option to login offline. Users that are not connected to the internet will be able to login by searching for the advertiser host, connect to its server and it will communicate with the actual login system, and will execute the login for him.