# PEMBUATAN *MULTIPLAYER GAME* UCING BELING MENGGUNAKAN *ASSET STORE MIRROR*

## *MAKING MULTIPLAYER GAME UCING BELING USING ASSET STORE MIRROR*

**Ryan Nanda Pratama[1], Anton Siswo Raharjo Ansori[2], Ashri Dinimaharawati[3]**

[1,2,3] Universitas Telkom, Bandung
**ryannp@student.telkomuniversity.ac.id**[1], **raharjo@telkomuniveristy.ac.id**[2],
**ashridini@telkomuniversity.ac.id**[3]

**Abstrak**

**Permainan tradisional merupakan permainan yang telah ada sejak dahulu dan wariskan turun-temurun pada generasi selanjutnya. Salah satu permainan tradisional yang berasal dari Jawa Barat adalah Ucing Beling. Permainan Ucing Beling saat ini sudah jarang dimainkan dan permainan tradisional ini kurang diminati masyarakat. Dengan berkembangnya permainan yang lebih modern dan menampilkan tampilan lebih menarik, membuat masyarakat lebih memilih memainkan permainan modern dan lambat laun permainan tradisional ini mulai dilupakan. Dalam penelitian ini akan melakukan perancangan pembuatan *muiltiplayer game* Ucing Beling menggunakan *Asset Store Mirror*. Yang mana penggunaan *mirror* dapat membuat sebuah *game* dimainkan oleh 2 orang atau lebih secara bersamaan dan diwaktu yang sama. Hasil dari penelitian tugas akhir ini penggunaan *Asset Store Mirror* dapat berjalan dengan yang diharapkan. Pembuatan fungsi *button* pada *TitleScreen Manager* dapat ditampilkan pada layar perangkat pemain dan dapat digunakan sesuai dengan fungsi yang diharapkan. Penggunaan *multiplayer* pada *game* Ucing Beling dapat dimainkan secara *realtime* antara pemain.**

**Kata kunci : *Game, Game Tradisional,* Ucing Beling*, Unity Asset Mirror, Multiplayer*.**

**Abstract**

**Traditional games are games that have existed since time immemorial and passed down generations in the next generation. One of the traditional games that originated in West Java is Ucing Beling. Ucing Beling games are rarely played and this traditional game is less popular with public. With the development of more modern games and featuring a more attractive look, making people prefer to play modern games and gradually this traditional game began to be forgotten. In this study will design the creation of ucing beling game muiltiplayer using Asset Store Mirror. Which is where use of mirrors can make a game played by 2 or more people simultaneously and at the same time. The results of this final task study use of Asset Store Mirror can run as expected. The creation of button functions in TitleScreen Manager can be displayed on screen of the player's device and can be used according to expected function. The use of multiplayer in Ucing Beling games can be played in realtime between players.**

**Keywords: Game, Traditional Game, Ucing Beling, Unity Asset Mirror, Multiplayer.**

### 1. Pendahuluan

Perkembangan teknologi di bidang *game* berkembang sangat cepat. Berbagai jenis *game* dapat kita temui dengan mudah, mulai dari *game online* maupun *offline*. Sebelum maraknya *game* pada zaman sekarang, dahulu permainan tradisional banyak dimainkan secara berkelompok oleh anak-anak. Permainan tradisional merupakan budaya yang telah diwariskan secara turun-temurun. Akan tetapi seiring berkembangnya teknologi yang semakin cepat, permainan tradisional mulai kurang diminati oleh anak-anak.

Permainan tradisional yang ada di Indonesia sangat beragam. Salah satunya adalah Ucing Beling. Ucing beling merupakan permainan tradisional yang berasal dari Jawa Barat. Permainan Ucing Beling saat ini sudah jarang dimainkan dan permainan tradisional ini kurang diminati masyarakat.

Agar menarik minat anak-anak dalam memainkan *game* Ucing Beling, penulis mengangkat topik tentang pembuatan *game multiplayer* ucing beling dengan *asset store mirror*. Penggunaan *asset mirror* guna pemain dapat memainkan *game* dengan 2 atau lebih pemain secara bersamaan dan dapat dimainkan secara terkoneksi dengan *internet* ataupun tidak.

## 2. Landasan Teori

### 2.1 Ucing Beling

Ucing Beling adalah salah satu permainan tradisional berasal dari Jawa Barat. Permainan yang dimainkan oleh 2 sampai 5 orang anak. Dalam permainan ini diharuskan mencari serpihan kaca yang telah disembunyikan oleh pemain. Serpihan kaca tersebut disembunyikan pada area tertentu yang diberi tanda dengan bentuk lingkaran atau garis batas tidak teratur. Dalam permainan ini, setiap anak membuat area dan menyembunyikan kaca, lalu mereka mencari serpihan kaca satu sama lain secara bergantian. Setiap anak atau pemain membuat suatu area dan menyembunyikan kacanya. Mereka mencari serpihan satu sama lain secara bergantian. Misalnya, A mencari kaca yang disembunyikan oleh B. Jika A menemukan pecahan kaca B, maka B akan melewati C. Jika B menemukan pecahan C, lalu C mencari pecahan kaca D, dan seterusnya, sampai orang terakhir mencari pecahan kaca orang pertama

### 2.2 *Multiplayer*

*Multiplayer* merupakan fitur pada *game* dimana pemain bermain dengan lebih dari 1 orang yang bermain di lingkungan *game* yang sama dan waktu yang bersamaan. *Game Multiplayer* biasanya memberikan pilihan pada pemain untuk berbagi sumber daya sistem game atau menggunakan *internet* untuk bermain bersama dalam jarak jauh. *Game Multiplaye* yang terhubung dengan *internet* melibatkan pemain yang saling terhubung melalui *server*. Sedangkan *Game Multiplayer* dengan koneksi lokal yaitu, pemain saling terhubung secara langsung dengan pemain lainnya, pemain terkoneksi menggunakan jaringan *peer to peer*. Pada *Game Multiplayer online* memiliki beberapa jenis kategori diantaranya adalah *Massively Multiplayer Online game* (MMO), *Massively Multiplayer Online First-person Shooter Games* (MMOFPS), *Massively Multiplayer online Real-time Strategy Games* (MMORTS), *Massively Multiplayer Online Role-playing Games* (MMORPG), *Multiplayer Online Battle Arena* (MOBA)[1]

### 2.3 Unity *Game Engine*

Penggunaan Unity dapat digunakan untuk mengembangkan konten interaktif seperti video game, visualisasi arsitektur, dan *real-time* 3D animasi. Pada Unity menggunakan bahasa pemograman JavaScript dan C#[2]. Unity *game engine* dapat digunakan dalam pembuatan dengan basis 2D dan 3D[3]. Pada Unity *game engine* mampu mengekspor *game* dengan basis kode yang sama pada berbagai *platform* antara lain: iOS, Android, Windows, Mac OS X, Linux, Playstation 4, Xbox One, dan web browser[4].

### 2.4 Unity *Asset Store*

Unity *Asset Store* adalah item yang dapat digunakan dalam *game* atau project. *Asset Store* dapat berasal dari file yang dibuat di luar Unity seperti, model 3D, file audio, gambar, atau jenis file lain yang didukung Unity. Ada beberapa jenis aset yang dapat dibuat dalam Unity, yaitu *Animator Controller*, *Audio Mixer*, dan *Render Texture*.

### 2.5 *Asset Mirror*

*Asset Mirror* merupakan sebuah fitur yang ada pada aplikasi *Unity* yang terdapat dalam Unity *Asset Store*. *Asset Mirror* memiliki hampir semua komponen dan fitur dari uNet, membuat jaringan mudah, ringkas, dan dapat dipertahankan. *Asset Mirror* memiliki berbagai komponen yang dapat ditambahkan kedalam objek game untuk memberi jaringan properti seperti pengidenfitikasian jaringan unik. Pada *Asset Mirror* telah memiliki *debugging* yang tersedia dalam *Unity logging system*. *Asset Mirror* bersifat *open-source* yang mana dapat diakses oleh siapa pun, dilihat, memodifikasi, dan mendistribusikan kode sesuai dengan keinginan pengguna[5].

### 2.6 C#

C# (C-sharp) adalah salah satu bahasa pemograman yang menggunakan Framework .NET. Sama seperti bahasa lainnya, C# memiliki aturan pada syntax dan kode-kode yang bisa digunakan dalam pembuatan aplikasi. C# cocok untuk dipelajari untuk pemula karena aturan *syntax*-nya lebih sederhana dibandingkan bahasa pemograman lainnya[6]. C# merupakan bahasa pemograman berorientasi objek. Diperkenalkan pertama kali pada bulan Juli 2000. C# merupakan bahasa pemograman berorientasi objek yang menjadi bahasa pemograman utama pada platform Microsoft.Net Framework. C# dianggap sebagai gabungan antara bahasa pemograman C++, pemograman *java*, serta penyederhanaan dari pemograman *Visual Basic*. Visual C# dapat ditemukan pada Microsoft Visual Studio[7]. C# juga mengusung konsep objek antara lain *inheritance, class, polymorphism* dan *encapsulation*[8].

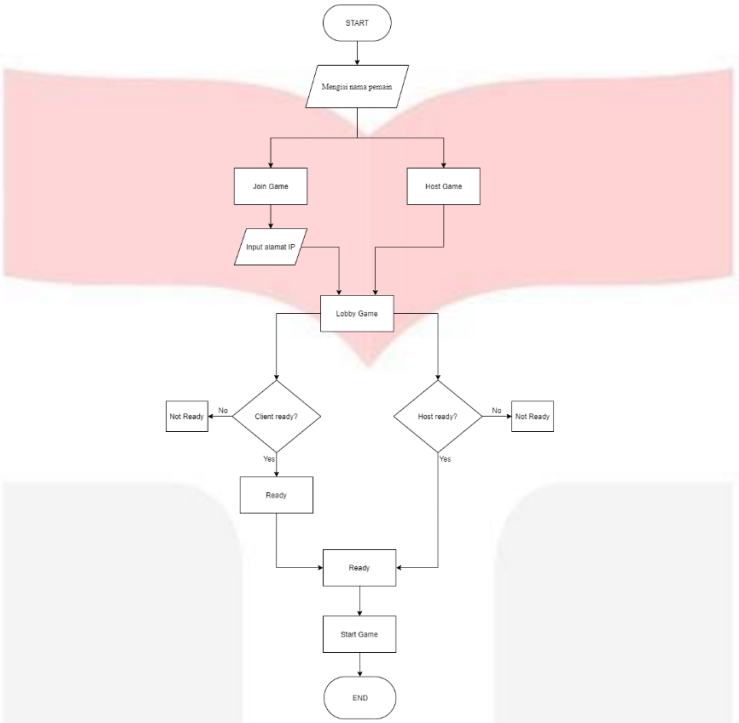
## 3. Metodologi Penelitian dan Perancangan

### 3.1 Desain Sistem

Sistem yang dibuat merupakan sebuah *game* berbasis desktop, *game* tradisional yang diadaptasikan dalam bentuk digital dengan cara mengintegrasikan bahasa pemograman dengan aplikasi *Unity*2D untuk pembuatan

objek permainan. Pada *game* akan menggunakan *Asset Mirror* yang tersedia pada *Asset Store* aplikasi *Unity*, yang nantinya *game* tersebut dapat dimainkan secara *multiplayer*. Fitur *Multiplayer* berfungsi untuk menjalankan sebuah permainan dengan 2 atau lebih pemain pada waktu bersamaan. Penggunaan *multiplayer* dapat bekerja dengan cara, pada tahap awal pemain akan mengisi nama terlebih dahulu. Selanjutnya pemain diberikan opsi *Host Game* dan *Join Game*. Pemain yang memilih *Host Game* akan bertugas sebagai *host* dan membuat *lobby game* secara otomatis oleh sistem. Pemain yang menjadi *host* dapat mengatur kapan permainan akan dimulai. Pemain yang memilih opsi *Join Game* akan diminta untuk memasukkan alamat *ip host*. Setelah itu akan bergabung kedalam *lobby game* yang telah dibuat oleh *host*. *Host* dapat memulai permainan ketika seluruh pemain dalam status *Ready*. Jika ada pemain yang berstatus *Not Ready*, maka *host* belum bisa memulai permainan. Pemain yang telah masuk didalam *lobby game* akan menunggu sampai *host* memulai permainan. Pada *lobby game* hanya akan ada 1 pemain yang menjadi *host*.

### 3.2 Diagram Blok



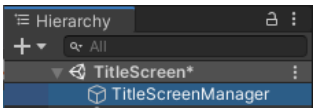

**Gambar 1.** Diagram Blok

Berdasarkan diagram yang telah dibuat, ketika pemain telah mengisikan nama pemain pada kolom yang disediakan, selanjutnya pemain diberikan 2 pilihan yaitu *Host game* dan *Join game*. Pemain yang memilih *host game* akan masuk kedalam *lobby game* yang telah dibuat secara otomatis oleh sistem dan pemain tersebut bertugas sebagai *host*. Pemain yang menjadi *host* pada *lobby game*, dapat mengubah status *not ready* menjadi *ready* dengan menekan tombol *ready up*. Jika pemain yang menjadi *host* dan pemain lain sudah dalam keadaan *ready*, maka tombol *start game* akan muncul pada layar *host*. Lalu jika ada salah satu pemain yang belum berstatus *ready*, maka pemain yang menjadi *host* harus menunggu sampai pemain lainnya sudah dalam keadaan *ready*. Pemain yang menjadi *host* dapat memulai *game* dan selanjutnya *game* akan muncul pada layar seluruh pemain dan *game* akan berjalan pada waktu yang sama. Pemain yang memilih opsi *Join game* selanjutnya pemain diminta untuk memasukkan alamat *ip host* pada kolom yang tersedia. Setelah pemain mengisi kolom alamat *ip host*, pemain akan memasuki *lobby game* yang berisikan *host*. Pemain dapat mengubah keadaan status *not ready* dengan menekan tombol *ready up* yang tersedia. Setelah mengubah status pemain menjadi *ready*, pemain akan menunggu hingga *host* memulai *game*.

### 3.3 Implementasi

#### 3.3.1 Implementasi Fungsi *TittleScreen Manager*

Dalam implementasi ini berfungsi untuk menampilkan *GameObject UI Panel* yang berisikan *button* dengan berbagai fungsi. Pada implementasi ini, penulis membuat sebuah *GameObject* yang diberi nama ”TitleScreenManager”.




**Gambar 2.** *GameObject*

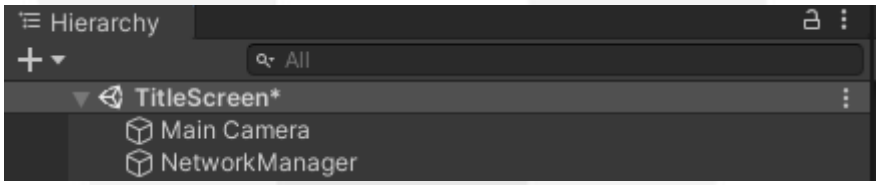
Berikut beberapa potongan kode yang telah disisipkan pada *GameObject*:

```
public class TitleScreenManager : MonoBehaviour
{
public static TitleScreenManager instance;
[SerializeField] private NetworkManagerUB networkManager;

[Header("UI Panels")]
[SerializeField] private GameObject mainMenuPanel;
[SerializeField] private GameObject PlayerNamePanel;
[SerializeField] private GameObject HostOrJoinPanel;
[SerializeField] private GameObject EnterIPAddressPanel;

[Header("PlayerName UI")]
[SerializeField] private TMP_InputField playerNameInputField;

[Header("Enter IP UI")]
[SerializeField] private TMP_InputField IpAddressField;

[Header("Misc. UI")]
[SerializeField] private Button returnToMainMenu;
public GameObject PopupsHowtoPlay;

private const string PlayerPrefsNameKey = "PlayerName";
```

Berdasarkan gambar potongan kode tersebut memiliki fungsi untuk membantu melabeli dan mengatur tampilan *UI* yang ada pada editor Unity.

### 3.3.2 Implementasi Implementasi Fungsi *Network Manager* Ucing Beling

Dalam implementasi ini, Fungsi *Network Manager* memiliki kendali inti dari pemainan dengan menggunakan *multiplayer*.

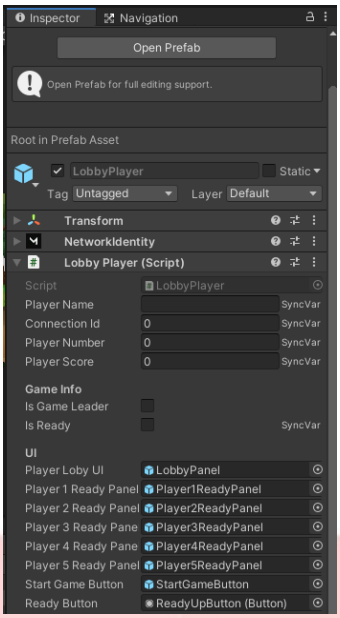



**Gambar 3.** *NetworkManager*

Pada implementasi ini, penulis membuat sebuah *GameObject* yang diberi nama ”*NetworkManager*”. Setelah *GameObject* tersebut dibuat, selanjutnya penulis menambahkan *component NetworkManager* yang tersedia pada *Asset Mirror*.

### 3.3.3 Implementasi Fungsi *Lobby Player*

Pada implementasi fungsi *Lobby Player* berfungsi agar para pemain bergabung kedalam *lobby*. Langkah pertama adalah membuat ”Player” *prefabs* yang akan dimunculkan oleh ”NetworkManager” pada saat *game* dibuat. ”NetworkManager” memang memiliki ”Player” *prefab* bawaan, yang akan ditambahkan oleh *player object* kosong.




**Gambar 4.** *LobbyPlayer*

### 3.3.4 Implementasi Fungsi *Game Player*

Pada implementasi Fungsi *Game Player* berfungsi ketika pemain telah memilih *button* ”HostGame” atau ”JoinGame”, dan pemain beserta *host* sudah dalam status *ready* maka pemain akan memulai *game* dan akan memuat *scene* ”Gameplay” pada layar seluruh pemain. *Prefab* ”GamePlayer” akan muncul oleh fungsi ”ServerChangeScene”, tetapi *prefab* tersebut akan muncul pada *scene* sebelumnya. Ketika *scene* berubah maka semua *scene* sebelumnya akan dihancurkan kecuali ”NetworkManager” karena ”NetworkManager” diberi label dengan nama ”DontDestroyOnLoad”. Supaya *prefab* ”GamePlayer” yang berisikan data pemain tidak hancur ketika *scene* berubah, *prefab* ”GamePlayer” ditandai dengan ”DontDestroyOnLoad”.

## 3.4 Pengujian

### 3.4.1 Pengujian Black Box

Pengujian Black Box dilakukan untuk memeriksa apakah tampilan, fungsi, dan tombol yang tersedia pada game sesuai dengan yang diharapkan .

a. Tabel hasil pengujian pada *panel* ”MainMenuPanel”. Pengujian yang dilakukan memeriksa fungsi tombol Start Game, How To Play, dan Exit Game.

**Tabel 1.** Hasil uji coba *panel* “MainMenuPanel”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol Start Game | Membuka panel "PlayerNamePanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |
| 2. | Tombol How To Play | Membuka *popups* "HowToPlay" | Valid | Berhasil berfungsi dengan yang diharapkan. |
| 3. | Tombol Exit Game | Keluar dari aplikasi *game*. | Valid | Berhasil berfungsi dengan yang diharapkan. |

b. Tabel hasil pengujian pada *panel* ”HostOrJoinPanel”. Pengujian yang dilakukan memeriksa fungsi tombol HostGame dan JoinGame.

**Tabel 2.** Hasil uji coba *panel* “HostOrJoinPanel”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol HostGame | Membuka panel "LobbyPanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |
| 2. | Tombol JoinGame | Membuka panel "EnterIPAddressPanel" | Valid | Berhasil berfungsi dengan yang diharapkan |

c. Tabel hasil pengujian pada *panel* ”PlayerNamePanel”. Pengujian yang dilakukan memeriksa fungsi tombol Enter Name dan Confirm.

**Tabel 3.** Hasil uji coba *panel* “PlayerNamePanel”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol Enter Name | - Membuka panel "HostOrJoinPanel" - Dapat memasukkan nama pemain | Valid | Berhasil berfungsi dengan yang diharapkan. |
| 2. | Tombol Confirm | Membuka panel "HostOrJoinPanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |

d. Tabel hasil pengujian pada *panel* ”EnterIPAddressPanel”. Pengujian yang dilakukan memeriksa fungsi tombol IPAddress dan JoinOnIP.

**Tabel 4.** Hasil uji coba *panel* “EnterIPAddressPanel”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol IPAddress | Dapat memasukkan alamat *ip*/nama host | Valid | Berhasil berfungsi dengan yang diharapkan. |
| 2. | Tombol Connect | Membuka panel "HostOrJoinPanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |

e. Tabel hasil pengujian pada *panel* ”ReturnToMainMenu”. Pengujian yang dilakukan memeriksa fungsi tombol Main Menu.

**Tabel 5.** Hasil uji coba *panel* “ReturnToMainMenu”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol Connect | Membuka panel "LobbyPanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |

f. Tabel hasil pengujian pada *panel* ”backToMainMenu”. Pengujian yang dilakukan memeriksa fungsi tombol Back.

**Tabel 6.** Hasil uji coba *panel* “backToMainMenu”

| No. | Pengujian | Hasil yang diharapkan | Hasil | Keterangan |
|---|---|---|---|---|
| 1. | Tombol Back | Membuka panel "MainMenuPanel" | Valid | Berhasil berfungsi dengan yang diharapkan. |

### 3.4.2 Pengujian *Multiplayer*

Pada pengujian *multiplayer game* Ucing Beling dimainkan secara langsung oleh 3 pemain. Dengan menggunakan 2 laptop dan 1 komputer.


**Gambar 5.** *Game* dimainkan oleh 3 orang pemain.

Pemain pertama setelah menekan tombol ”Start Game”, pemain memasukkan nama”Ari”.


**Gambar 6.** Pemain pertama memasukkan nama “Ari”

Setelah pemain pertama menekan tombol ”Confirm”, muncul opsi ”HostGame” dan ”JoinGame”. Pemain pertama memilih opsi ”HostGame” selanjutnya masuk kedalam *lobby*, pemain pertama akan bertugas sebagai *host* karena pemain memilih opsi ”HostGame”.


**Gambar 7.** Ari memilih opsi “HostGame”

Setelah pemain kedua menekan tombol “StartGame”, pemain kedua memasukkan nama “Ryan”. Setelah menekan tombol “Confirm”, Pada layar laptop muncul opsi “JoinGame” dan “HostGame”. Pemain kedua memilih opsi “JoinGame”.


**Gambar 8.** Pemain kedua memasukkan nama “Ryan”

Selanjutnya pemain kedua diminta untuk memasukkan alamat *ip host* sebelum bergabung kedalam *lobby.* Karena *host* pada *lobby* adalah pemain pertama dengan nama “Ari”, maka pemain kedua dapat memasukan alamat *ip* pemain pertama. Alamat *ip* yang dimasukkan adalah versi IPv4 pada komputer yang digunakan oleh *host* dengan alamat: 170.20.10.2.


**Gambar 9.** Pemain kedua memasukkan alamat *ip host*

Setelah pemain kedua memasukan alamat *ip host*. Selanjutnya pemain menekan tombol ”Connect” untuk masuk kedalam *lobby* dan terhubung dengan *host*. Setelah terhubung dengan *host*, pemain pertama dan kedua berada pada *lobby* yang sama.

Selanjutnya pemain ketiga, setelah menekan tombol “StartGame” pemain memasukkan nama “Arivo”.


**Gambar 10.** Pemain ketiga memasukkan nama “Arivo”

Setelah menekan tombol “Confirm”, Pada layar laptop muncul opsi “JoinGame” dan “HostGame”. Pemain ketiga memilih opsi “JoinGame”. Selanjutnya pemain ketiga diminta untuk memasukkan alamat *ip host* sebelum bergabung kedalam *lobby.* Maka pemain ketiga memasukkan alamat *ip host* seperti yang dilakukan pemain kedua.


**Gambar 11.** Ketiga pemain telah berada pada *lobby* yang sama

Setelah memasukkan alamat *ip host*, pemain ketiga menekan tombol "Connect" untuk terhubung kedalam *lobby* yang berisikan *host*. Selanjutnya pemain ketiga akan bergabung bersama *host* dan pemain kedua. Setelah seluruh pemain berada pada *lobby*, seluruh pemain mengubah status "Not Ready" menjadi "Ready" dengan menekan tombol "Ready Up" untuk memulai permainan.


**Gambar 12.** Ketiga pemain telah berstatus "Ready"

Ketika seluruh pemain sudah berstatus "Ready", pada layar pemain pertama yang bertugas sebagai *host* akan muncul tombol "Start Game". Tombol tersebut berfungsi untuk memulai *game* bersama dengan pemain lain.

**Gambar 13.** *Game* muncul pada layar ketiga pemain

## 4. Kesimpulan

Berdasarkan proses pengerjaan pengujian dan analisis yang telah dilakukan pada tugas akhir ini, maka dapat ditarik kesimpulan bahwa:

1. Pembuatan fitur *multiplayer* pada *game* Ucing Beling dengan menggunakan *Asset Store Mirror* dapat berjalan sesuai dengan yang diharapkan.
2. Masing-masing tombol pada *panel* yang dinaungi oleh *canvas* pada *TitleScreen Manager* dapat ditampilkan dan digunakan sesuai dengan fungsi yang diharapkan.
3. Berdasarkan pengujian *Multiplayer*, *client* dan *host* dapat terhubung sesuai dengan pembuatan *LobbyPlayer*.
4. Pada *multiplayer game* Ucing Beling dapat dimainkan secara *realtime* dan berjalan dengan sesuai yang diharapkan.

## Referensi :

[1] Gede Humaswara Prathama, N. M. Ary Esta Dewi Wirastuti, Y. Divayana, "Analisa Penggunaan WebRTC dan Websocket pada Real Time Multiplayer Online Game Tradisional Ceki", Majalah Ilmiah Teknologi Elektro, Vol. 18, No. 1, Januari 2019.

[2] W. Huang, H. Xiang, Shaohul Li, "The application of augmented reality and unity 3D in interaction with intangible cultural heritage", Springer, November 2019.

[3] Pratik P. Patil, dan Ronal Alvares, "Cross-platform Application Development using Unity Game Engine", International Journal of Advance Research in Computer Science and Management Studies, April 2015.

[4] S. S. Sarwodi, W. S. Wardhono, M. A. Akbar, " Penerapan Multiplayer Pada Gim Tower Defense Menggunakan Photon Unity", Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, September 2020.

[5] Andreas Lindblom, "A Study of Networking Performance in a Multi-user VR Environment", Thesis, Lulea University of Technology, Swedia, 2020

[6] Jubilee Enterprise, *Belajar Sendiri Visual C# dan C++ untuk Pemula*. Jakarta: PT Elex Media Komputindo, 2016, pp. 3.

[7] L. Hakim, *Pemograman C# dan EmguCV: PENGENALAN C# DALAM VISUAL STUDIO*. Yogyakarta: Deepublish, 2018, pp. 11.

[8] G. S. Paruntu, S. T. Godion Kaunang, V. Tulenan, "Game Based Education: Shorinji Kempo", Jurnal Teknik Informatika, Vol. 15, No. 2, April 2020.