🃏 Multiplayer Card Game – Unity Assignment

Githib Repo Link: https://github.com/YesAvi90/AlterOffice_Assignment

📌 Overview

This project is a 1v1 online multiplayer card game built in Unity using Mirror networking.

Two players connect over LAN (Host–Client model) and compete across 6 rounds.
Each round consists of:

- Card selection phase (30s timer)
- Simultaneous turn submission
- Alternating reveal phase
- Score resolution
- Ability resolution

The winner is determined after 6 rounds based on total score.

🛠️ Tech Stack

- **Engine:** Unity (2022.3.62f2)
- **Networking:** Mirror (Host–Client architecture)
- **Language:** C#
- **Data:** JSON-based card configuration
- **Target Platform:** Android (APK included)

🎮 Core Gameplay Rules

🔹 Match Flow

- Total rounds: **6**
- Each round:
    - Players receive energy equal to the round number
    - Players select cards within 30 seconds
    - When both players end their turn, → The reveal phase begins

---

🔹 Initiative System

Initiative determines which player reveals first each round.

Calculation:

1. The player with the higher total score goes first
2. If tied → random selection

Reveal order alternates per card during the reveal phase:

- The initiative player reveals first
- Opponent reveals the second
- Repeat until all cards are revealed

🔹 Energy System

- Round 1 → 1 energy
- Round 2 → 2 energy
- …
- Round 6 → 6 energy

Players cannot select cards exceeding available energy.

✨ Implemented Card Abilities

The game includes the following abilities:

1️⃣ StealPoints

Steals 2 points from opponent (if available).

2️⃣ BonusPoints

Adds bonus points equal to the number of cards played that turn.

3️⃣ DrawCard

Draws 1 additional card from the deck.

4️⃣ DiscardOpponentCard

Removes a random previously played card from the opponent's history and subtracts its power from their score.

🌐 Networking Architecture

🔹 Multiplayer Model

- The host acts as an authoritative server.
- Client connects via IP address.
- Maximum 2 players.

🔹 Event-Driven Messaging

All gameplay communication uses JSON messages with an `action` field.

Example structure:

```
{
  "action": "RevealCard",
  "payload": { ... }
}
```

No primitive RPC data is sent directly.
Game state transitions are triggered via structured messages.

📂 Card Data (JSON Driven)

Cards are loaded from a JSON file structured like this:

```
{
  "cards": [
    {
      "id": 1,
      "name": "Shield Bearer",
      "cost": 2,
      "power": 3,
      "ability": {
        "type": "None",
        "value": 0
      }
    }
  ]
}
```

This allows easy balancing and extensibility without modifying core logic.

▶ How to Run (Unity – LAN Multiplayer)

Requirements

- Two devices (PC or Android)
- Both devices must be connected to the **same local network (e.g., same Wi-Fi)**

Steps

1. On Device 1 → Click **Host**
2. The Host's IP will be displayed on the screen.
3. On Device 2 → Enter the Host's local IP address and click **Join**

⚠️ Troubleshooting

If the client is unable to connect:

- Ensure both devices are on the **same Wi-Fi / LAN network**
- Verify the **Host's local IP address** is correct
- Try temporarily disabling:
    - Windows Firewall
    - Antivirus network blocking
    - Other security software that may block local connections

After successful testing, firewall settings can be restored.

📱 Android Build

- APK included in submission
- Build target: Android
- Tested on a physical Android device

⚠️ Known Limitations

- Reconnection support was explored but not fully implemented due to the 6-day development constraint.
- If a player disconnects, the match resets and returns to matchmaking.

🧠 Architectural Notes

The project emphasizes:

- Clear separation of:
    - Networking layer

    - Turn management
    - UI
    - Card logic
- Server-authoritative score and reveal resolution
- Modular ability handling
- JSON-driven card configuration

👤 Developer

Yash Avinash
Unity Developer Candidate