📘 네트워크게임 작업내역서_이무준

프로젝트명: IP Trade Simulator

1. 프로젝트 개요

- **프로젝트명:** IP Trade Simulator
- **개발 목적:**
  - 플레이어 간 **순간 판단력 + 순발력 경쟁**을 만드는 네트워크 게임
  - 단순 입력 경쟁이 아닌 **정보전·심리전·추리 기반의 재미** 구현
- **구성:** Unity 클라이언트 + C#(.NET) 서버
- **네트워크:** 서버 권한 기반(Authoritative Server)
- **핵심 루프:**
  i. 정답(숫자) 추리
  ii. 로그(힌트) 공유
  iii. 상점 라운드
  iv. 역전 가능성
  v. 라운드 반복

2. 게임 기획 의도

2-1. 정보 공유 기반 심리전 강화

플레이어가 입력한 숫자와
그 숫자가 **정답보다 높은지/낮은지**의 모든 정보가
실시간으로 모든 플레이어에게 공유된다.

이를 통해 단순한 숫자 입력 게임이 아니라

- 누가 범위를 좁히고 있는지

- 누가 위험한 선택을 하는지
- 누가 정답에 근접했는지

이 모든 정보가 서로에게 영향을 주는 **심리전 구조**로 발전한다.

2-2. 반복 플레이의 신선함을 유지하는 랜덤 상점

매 라운드 상점은 완전히 새롭게 생성된다.

- 아이템 이미지
- 아이템 이름 (prefix + Very*0~2 + baseName 조합)
- 가격 공식
- 실제 가격

이 모두 랜덤으로 출력되며,
단순한 규칙이지만 조합 확장성이 매우 커서
매판 다른 플레이 경험을 제공한다.

2-3. 제한시간 + 계산식 가격을 통한 긴장감 유지

상점 이용에는 **시간 제한**이 있으며,
가격 또한 계산식("12*3 + 5") 형태라
빠른 계산 + 판단력 + 순발력이 동시에 요구된다.

정답을 먼저 맞추었다고 해서 끝이 아니라,

> **“상점에서 누가 더 빠르고 정확하게 판단하나?”**

이 경쟁이 한 번 더 발생하도록 설계했다.

2-4. 선구매 이후에도 끝까지 유지되는 역전 시스템

가장 먼저 상점에서 아이템을 구매해도 **라운드가 즉시 종료되지 않는다.**

- 첫 구매 이후 **30초 후에 라운드 종료**

- 그 사이 다른 플레이어가 더 비싼 상품을 구매하면 **역전 가능**

즉,
✓ 정답을 가장 먼저 맞춘 사람이
✓ 상점에서 가장 큰 이득을 본 사람이
✓ 라운드에서 최종적으로 승리한 사람이

모두 다를 수 있게 설계했다.

이를 통해 **마지막 30초까지 긴장감이 유지되는 구조**를 만들었다.

3. 기능 구현 작업 내역

3-1. 로그(추리 힌트) 공유 시스템

구현 내용

- 서버가 모든 플레이어의 입력을 집계
- 정답과 비교하여 high/low 판정
- 모든 플레이어에게 동일한 로그 실시간 전송
- InfoPanelManager에서 UI 갱신

개발 중 문제 & 해결

- 스냅샷 갱신이 UI 중복 갱신을 유발
  → UI 갱신 함수 단일화, 이전 단계 레이스 컨디션 제거
- 정답 비교 시 int/string 혼용 문제
  → DTO 타입 구조를 명확히 분리해 해결

3-2. 랜덤 상점 시스템

구현 내용

- Seed 기반 랜덤 아이템 생성
- IconPool에서 sprite명 추출 → 아이콘 규칙 통합

- prefix/very/baseName 조합으로 이름 생성
- 가격 공식과 실제 가격 생성
- 상점 아이템 구매 시 서버 측 처리 후 스냅샷 갱신

개발 중 문제 & 해결

- sprite명 정규화 과정에서 숫자 제거 시 prefix까지 삭제
  → 아이콘 키 정규화 메서드 별도 분리
- UI 풀링 과정에서 이벤트가 중복 발생
  → View Setup 분리 및 풀링 구조 개선

3-3. 제한시간·타이머·역전 시스템

구현 내용

- 첫 구매 발생 시 서버에서 TimeUntil 설정
- 30초 카운트다운 후 자동 라운드 종료
- 종료 전 추가 구매는 모두 유효

문제 & 해결

- 스냅샷 주기에 따라 간헐적으로 TimeUntil이 튀는 문제
  → “서버 최신 값 우선 적용” 규칙 도입

3-4. 라운드 진행 구조

구현 내용

- Waiting → Playing → Shop → EndRound → Repeat
- 정답 생성, 로그 공유, 상점 열기 등 모든 진행은 서버 통제
- 클라이언트는 스냅샷 기반 UI 갱신

문제 & 해결

- 라운드 전환 시 UI 업데이트 타이밍이 늦는 문제
  → Phase 전환 시점에 Explicit UI Refresh 추가하여 해결

4. 네트워크 기반 게임 구조 설계 (요약)

```
[SERVER]
 ├ 플레이어 상태 관리
 ├ 라운드 제어
 ├ 정답 판정 및 로그 전달
 ├ 상점 아이템 생성
 ├ 구매 처리
 └ 스냅샷 전송 (TCP)

[CLIENT]
 ├ 입력 전송
 ├ 스냅샷 수신 후 UI 갱신
 ├ 상점 UI/구매 기능
 ├ 제한시간 표시
 └ 로그 UI 처리
```

5. 회고

이번 프로젝트의 목표는 **“네트워크 기반 환경에서 매 라운드 긴장감이 살아있는 게임을 완성하는 것”**이었다.

이를 위해 다음 요소들을 결합했다:

- 정보 공유에 따른 심리전
- 랜덤 요소를 통한 매판 신규성
- 제한시간으로 인한 압박감
- 상점에서의 계산 및 판단 경쟁
- 선구매 이후에도 이어지는 역전 구조

단순한 형태의 게임이지만,
플레이어의 선택과 추리가 주는 재미를 극대화할 수 있었다.