

결 재	담당	원장

자바 스프링 리액트로 완성하는 클라우드 활용 풀스택 개발

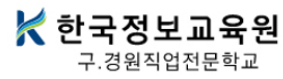
1차 프로젝트 완료 보고서

아수라(ASURA)

2025.07.28

구성원 : 김기찬
조유빈
이영호

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28



프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

목차

1. 프로젝트 개요	
● 프로젝트 명	4
● 프로젝트 기간	4
● 프로젝트 목표	4
● 프로젝트 시나리오	4
● 프로젝트 수행 요건	4
2. 프로젝트 추진 체계	
● 프로젝트 참여 인력 총괄표	5
● 참여 인력 업무 분장	5
3. 세부 프로젝트 내용	
● 전체 구성도	6
● 네트워크 구성도	7
● 서버(물리/가상) 구성 현황	10
● 네트워크 구성 현황	12
● 상세 구축 및 구성 내용	14
● 구축 결과	16
4. 개별 후기	17

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

1. 프로젝트 개요

- 프로젝트 명
아수라(ASURA)
- 프로젝트 기간
2025.07.18.- 2025.07.28.
- 프로젝트 목표
 - 가) 웹 브라우저에서 설치 없이 바로 즐길 수 있는 PvP 전투 게임을 구현
 - 나) 2~8인 멀티플레이 전투, 다양한 무기 및 모드를 제공하는 실시간 액션 게임 제작
 - 다) Three.js, Socket.IO, Node.js 기반의 웹 3D 게임 환경 구축
 - 라) 향후 플랫폼 확장을 고려한 모듈형 구조 설계
- 프로젝트 시나리오
 - 가) 가정
 - 본 프로젝트는 설치가 필요 없는 웹 기반 실시간 PvP 게임을 구현하는 것을 목표로 하며, 3D 렌더링, 실시간 네트워크 통신, 서버-클라이언트 구조 등 실무 중심의 기술을 직접 설계 및 구현한다는 가정을 기반으로 한다.
 - 전체 시스템은 서버-클라이언트 이중 구조로 운영되며, 서버는 Node.js + Express.js 기반으로 구성, 클라이언트는 Three.js 기반의 3D 환경을 렌더링한다.
 - 데이터베이스 없이 메모리 기반 상태 관리를 통해 빠른 응답성과 단순 구조를 유지하고, Socket.IO를 활용한 양방향 실시간 통신으로 전투 이벤트, 플레이어 이동, 피격, 사망 등의 동기화를 수행한다.
 - 나) 대상
 - 브라우저에서 설치 없이 게임을 즐기고자 하는 캐주얼 게이머
 - 다) 세부내용
 - 게임은 웹 브라우저에서 실행되며, 사용자는 캐릭터 선택 후 실시간 전장에 입장하여 전투를 수행한다. 전투는 다음과 같은 구성 요소로 진행된다:
 - ◇무기는 근접(검, 도끼 등) 및 원거리(총기, 활 등)로 구분
 - ◇각 무기는 공격력, 공격 범위, 속도에 따라 밸런스 조정
 - ◇무기 장착/해제, 투사체 생성, 애니메이션 연동, 히트 판정 등을 포함
 - ◇기본 대전(1:1, 자유전), 팀전, 점프맵, 보스 레이드, 1인칭 모드
 - ◇다양한 맵 (도시, 무인도, 당구대, 우주 등) 제공
 - ◇각 모드는 플레이어 수와 전투 방식에 따라 룰과 승패 조건이 달라짐
 - ◇Three.js로 클라이언트에서 3D 씬, 캐릭터, 무기 렌더링 수행
 - ◇Socket.IO로 실시간 위치, 상태, 공격 등의 통신 처리
 - ◇서버는 Room 단위의 세션 관리와 게임 상태 처리 담당

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- 프로젝트 수행요건
 - HTML, CSS, JavaScript 기반의 클라이언트 UI 및 렌더링 구성
 - Three.js로 3D 그래픽 렌더링 구현
 - Node.js + Express.js로 서버 환경 구성 및 정적 리소스 제공
 - Socket.IO로 실시간 통신 구현 및 클라이언트 간 이벤트 동기화
 - Render를 이용한 웹 배포 환경 구축
 - 최소 클라이언트 요구 환경: Chrome 브라우저 (데스크탑 기준)
 - 모든 시스템은 별도 DB 없이 메모리 기반 동작으로 구성됨

2. 프로젝트 추진 체계

- 프로젝트 참여 인력 총괄표

성명	소속	역할	담당업무
이영호	대진대학교	Project Leader	전투 시스템 구현, 보고서 작성, 아이템 시스템 구현, 시각 효과
김기찬	대진대학교	Project Assistant	서버 구축, 코드 안정화, 캐릭터 모션 시스템 구축, 맵 로직 및 렌더링 구현
조유빈	대진대학교	Project Assistant	UI 설계 및 구현, 맵 로직 및 렌더링 구현

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

● 참여 인력 업무 분장

업무명	업무내용
보고서 작성	<ul style="list-style-type: none"> - 프로젝트 기획서 작성 - 프로젝트 결과 보고서 작성 - 프로젝트 진행 상황에 대한 일정 조정 - 기타 보고서 작성 및 발표 진행
전투 시스템 구현	<ul style="list-style-type: none"> - 근접무기(검, 도끼, 단검, 망치) 및 원거리무기(총기, 활) 분류 체계 구축 - 무기별 데미지, 공격 범위, 공격 속도 등 밸런스 조정 - 무기 장착/해제 시스템 구현 - 무기 픽업 메커니즘 개발 - 플레이어 공격 애니메이션 시스템 (SwordSlash, Shoot_OneHanded, walkAttack) - 복합모션 시스템 (상체/하체 애니메이션 분리) - 공격 판정 및 히트박스 구현 - 피격 시 시각적 피드백 (빨간 화면 효과, 체력바 감소) - 플레이어 체력 관리 시스템 - 사망 및 리스폰 메커니즘 - 맵 경계 데미지 시스템 (맵 밖으로 나갈 시 지속 데미지) - 사망 시 카운트다운 UI 및 부활 타이머
아이템 시스템 구현	<ul style="list-style-type: none"> - FBX 형식의 무기 모델 로더 구현 - 무기별 스케일링 및 위치 조정 (총기류 0.005, 근접무기 0.01) - 캐릭터 본(Bone)에 무기 부착 시스템 - 무기별 공격 모션 연동 - 무기 회전 및 위치 조정 (근접무기 Y축 90도 회전 등) - 공격 시 무기 모션 동기화
시각 효과	<ul style="list-style-type: none"> - 플레이어 피격 시 빨간 화면 오버레이 효과 - 페이드 인/아웃 트랜지션 효과 - 사망 시 화면 어둡게 처리 및 메시지 표시 - HP바 애니메이션 및 색상 변화 - 캐릭터 선택 시 네온 글로우 효과 - 버튼 호버 및 클릭 애니메이션

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

업무명	업무내용
서버 구축	<ul style="list-style-type: none"> - HTTP 서버 초기화 및 설정 - 정적 파일 서빙 시스템 (public 폴더) - 라우팅 시스템 구현 - 포트 설정 및 환경 변수 처리 - 단일 서버 구조로 게임 로직 통합 관리 - 메모리 기반 게임 상태 관리 - 서버 성능 최적화 및 메모리 관리 - Socket.IO 서버 초기화 및 설정 - 클라이언트-서버 간 양방향 통신 구현 - 이벤트 기반 메시징 시스템 - 방(Room) 생성 및 관리 시스템 - 플레이어 입장/퇴장 처리 - 게임 상태 실시간 동기화 - 네트워크 지연 처리 및 최적화
캐릭터 모션 시스템 구축	<ul style="list-style-type: none"> - Three.js AnimationMixer 기반 모션 시스템 - 캐릭터별 애니메이션 클립 관리 - 애니메이션 블렌딩 및 트랜지션 처리 - 멀티플레이어 애니메이션 동기화 - 이동, 점프, 구르기, 공격 애니메이션 연동 - 상체/하체 애니메이션 분리 시스템 - 애니메이션 상태 머신 구현
맵 로직 및 렌더링 구현	<ul style="list-style-type: none"> - WebGL 렌더러 초기화 및 설정 - 안티앨리어싱 및 그림자 매핑 시스템 - 렌더링 품질 최적화 (sRGB, 감마 보정) - 방향성 조명 및 반구형 조명 설정 - 동적 그림자 시스템 구현 - 하늘 및 안개 효과 셰이더 구현 - Three.js 라이브러리 통합 및 설정 - 씬(Scene), 카메라, 렌더러 초기화 - 3D 모델 로더 (GLTF, FBX) 구현 - 텍스처 및 머티리얼 시스템 구축 - 탐류 카메라 구현 - 플레이어 추적 카메라 시스템 - 화면 비율 대응 및 반응형 처리
UI 설계 및 구현	<ul style="list-style-type: none"> - 26개 캐릭터 그리드 레이아웃 구현 - 캐릭터 썸네일 및 3D 미리보기 시스템 - 네온 글로우 효과 및 선택 애니메이션 - 반응형 디자인 및 모바일 패치 - 전투 게임 테마의 시각적 디자인 - 캐릭터 카드 호버 효과 및 트랜지션 - 닉네임 입력 및 유효성 검사

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

업무명	업무내용
HP 시스템	<ul style="list-style-type: none"> - 플레이어 HP 초기화 및 관리 (기본 100) - 데미지 적용 및 체력 감소 처리 - 사망 판정 및 상태 변경 로직 - 리스폰 시스템 및 체력 복구 - hp.js 기반 3D 공간 HP바 구현 - 플레이어 머리 위 HP바 표시 - 실시간 HP 변화 애니메이션 - 캐릭터별 프로필 이미지 연동 - 사망 시 카운트다운 오버레이 표시 - 리스폰 타이머 및 위치 재설정 - 사망 상태에서 입력 비활성화 - 부활 시 상태 초기화 및 무기 해제

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

3. 세부 프로젝트 내용

- 시스템 아키텍처

본 프로젝트는 클라이언트-서버 구조 기반의 실시간 PvP 게임으로 설계되었다.
클라이언트는 브라우저 환경에서 실행되며, **Three.js**를 사용한 3D 렌더링과 사용자 입력 처리, UI 인터페이스를 담당한다.

서버는 **Node.js + Express.js + Socket.IO** 를 기반으로 구성되며, 게임 매칭, 상태 관리, 공격 판정 등의 로직을 처리하고 실시간으로 위치, 상태, 이벤트 정보를 관리한다.

게임 구조는 다음 세 가지 구성요소로 나뉜다:

클라이언트 (Client):

HTML, CSS, JavaScript, Three.js 기반 3D 씬 렌더링, UI 및 게임 씬 구현
사용자 입력과 애니메이션, UI 상태를 처리하고 서버와의 실시간 통신을 통해 동기화

서버 (Server):

Node.js와 Express.js 기반으로 구현.
실시간 게임 룸 관리 및 상태 계산 수행
충돌 판정, 공격 처리, 생존 여부 등 핵심 로직 담당
일정 주기마다 게임 상태를 모든 클라이언트에 전파

통신 (Socket.IO):

실시간 플레이어 위치, 공격, 피격, 사망 등의 이벤트를 양방향으로 전송.
Room 단위 통신 구조로 세션을 구분하고, 브로드캐스트 방식으로 동기화 처리.

통신 구조 및 실시간 처리 방식

- 명령 이벤트 기반 통신

- 클라이언트와 서버는 '입장', '이동', '공격', '피격', '사망' 등 명령 기반으로 통신
- 각 이벤트는 특정 포맷으로 구조화되어 송수신됨

- Tick 기반 게임 루프

- 서버는 약 30ms 주기로 전체 게임 상태를 재계산
- 클라이언트는 서버에서 받은 상태를 보간하여 자연스럽게 화면을 갱신

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- 서버 절대 기준 판정 체계
- 위치, 공격 등 모든 판정은 서버 기준
- 클라이언트는 조작만 입력하고, 결과는 서버가 최종 판단

시스템 구성도

[Client (Browser)]

|

| Socket.IO (WebSocket)



[Game Server (Node.js)]

|

├─ Game State Manager (Object 위치/상태 추적)

├─ Physics Simulation (충돌 판정, 무기 판별)

└─ Matchmaking / Room Handler

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

클라이언트-서버 간 데이터 흐름

흐름 단계	설명
초기 접속	클라이언트는 서버에 Socket 연결을 요청하며 접속 이벤트 발생
방 생성/참가	서버는 Room ID를 발급하거나 기존 Room에 플레이어 추가
게임 진행	서버는 매 틱(30~60fps)마다 오브젝트 상태를 브로드캐스트
이벤트 처리	클라이언트 공격/스킬 입력은 서버에 전달 후 판정 결과 반영
종료 처리	게임 종료 시 점수 및 플레이어 상태 서버에 저장 또는 종료 처리

기술적 고려사항

- 실시간성 확보 전략

- WebSocket 기반의 Socket.IO 채널을 Room 단위로 구성하여 개별 게임 세션 분리
- 위치 정보는 서버가 신뢰하는 Authoritative Server Model 채택
- 상태 업데이트 간 간섭 방지를 위해 Tick 기반 스냅샷 동기화 모델 구현

- 서버 부하 및 확장 가능성

- 현재는 단일 Node.js 서버 구조이나, 향후 **Redis Pub/Sub, Socket.IO Adapter** 기반으로 수평 확장이 가능하도록 설계
- 방(Room) 단위 분산 구조로 확장 시에도 사용자간 데이터 충돌 없이 병렬 처리 용이

- 보안 및 오류 처리

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- 클라이언트 조작 방지를 위해 핵심 게임 로직은 서버 측에서 처리
- 연결 해제, 중복 연결 등 예외 케이스 대응 로직 포함
- Match 종료 시, Room 자원 자동 정리 및 사용자 소켓 해제

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

● 기술 구성 및 기능별 구현

- **Three.js**를 활용해 3D 캐릭터, 맵, 무기 렌더링 구현
- **Socket.IO** 기반의 실시간 플레이어 동기화 기능 구현
- **Express.js**로 라우팅 및 정적 파일 서빙 처리
- **Render** 플랫폼을 통해 클라우드 배포 진행

기능 구현은 다음과 같은 항목을 중심으로 진행되었다:

캐릭터 및 맵 구현

- 총 26종의 캐릭터 선택 가능
- 도시, 무인도, 당구대, 우주 등 다양한 맵 테마 제공

무기 시스템

- 근접/원거리 무기로 분류 (검, 도끼, 활, 총기 등)
- 무기별 데미지, 사거리, 속도, 애니메이션 연동
- 투사체 처리 및 충돌 판정 구현

게임 모드 및 전투 구성

- 개인전(1:1 자유 대전) 모드 우선 구현
- 향후 팀전, 점프맵, 레이드 등 확장 예정

UI/UX 및 인터랙션 요소

- 캐릭터 선택 UI, HP바 애니메이션, 사망 시 인터페이스 처리
- 네온 효과, 모달 팝업 등 시각적 효과 적용

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

기술 구현 세부 항목

- Three.js 주요 컴포넌트 활용 방식 (Scene, Camera, Renderer, Light)
- 3D 모델 로딩 및 애니메이션 처리 (GLTFLoader, AnimationMixer)
- 충돌 판정 알고리즘 (AABB, Raycasting 등 적용 여부)
- 캐릭터/무기 애니메이션 분리 및 전환 로직
- 서버-클라이언트 상태 동기화 방식 (tick 기반 vs event 기반)
- Socket.IO 커스텀 이벤트 목록 및 흐름 설명
- 투사체 구현 방식 (physics-based vs 직선 이동 등)

네트워크 통신 및 실시간 동기화

- 통신 프로토콜 및 방식

- WebSocket 기반 Socket.IO를 활용하여 클라이언트와 서버 간 실시간 양방향 통신을 구현하였다.
- 각 클라이언트는 서버 접속 후 특정 게임 룸(Room)에 참가하며, 룸 단위로 세션이 분리되어 관리된다.

- 명령 이벤트 기반 메시징

- 클라이언트와 서버는 '입장', '이동', '공격', '피격', '사망' 등 주요 게임 이벤트를 명령(event) 형태로 송수신한다.
- 모든 이벤트는 JSON 포맷으로 구성되며, 서버는 이를 검증 및 처리한다.

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- Tick 기반 게임 루프

- 서버는 약 30ms(초당 30~33회) 간격으로 고정 주기 Tick 루프를 운영한다.
- 각 Tick마다 플레이어 위치, 공격 상태, 충돌 판정 등 핵심 게임 로직을 순차 처리하며, 업데이트된 상태를 룸 내 모든 클라이언트에 브로드캐스트한다.

- 클라이언트 상태 보간

- 클라이언트는 서버로부터 받은 Tick 간격 상태 정보를 보간(Interpolation)하여 위치 및 회전 변화를 부드럽게 렌더링한다.
- 이로 인해 네트워크 지연이나 불규칙한 데이터 수신으로 인한 화면 끊김 현상을 최소화한다.

- 서버 권위(Authoritative) 모델

- 게임의 최종 상태와 판정은 서버가 관리하며, 클라이언트는 입력 명령 전송에 집중한다.
- 충돌, 공격 판정 및 게임 결과 모두 서버 기준으로 처리하여 공정성을 보장한다.

- 지연 및 오류 대응

- 네트워크 지연(Ping) 측정과 시간 보정 알고리즘을 적용하여 동기화 문제를 완화하였다.
- 비정상 데이터 수신 시 이벤트 무시 및 재요청 메커니즘을 통해 안정성을 확보했다.

코드 구조 및 품질 관리

코드 모듈화

- 클라이언트 측 코드는 player.js, weapon.js, attackSystem.js 등 역할별 모듈로 분리하여 유지보수를 용이하게 설계하였다.
- 서버 측 코드는 index.js, gameRoomManager.js, eventHandler.js 등 기능 단위로 모듈화하였다.

테스트 및 성능 평가

- 렌더링 및 프레임률 저하 원인을 분석하고 최적화 작업을 반복 수행하였다.

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- 네트워크 지연 및 패킷 손실 상황을 시뮬레이션하여 클라이언트 측 예측 및 보간 기법의 효과를 검증했다.
- 동시 접속 플레이어 수에 따른 서버 부하 테스트 및 안정성 검증을 완료하였다.

디버깅 및 문제 해결 사례

- 동기화 오류

- 문제: 캐릭터 위치 불일치, 클라이언트 공격이 서버에서 무효 처리
- 원인: Tick 간 시간차, 네트워크 지연, 부적절한 보간 방식
- 조치: Ping 기반 시간 보정, 보간 방식 개선, Tick offset 로그 추가

- 충돌 판정 오류

- 문제: 근접 공격 미명중, 반대로 멀리서 피격
- 원인: 히트박스 위치 계산 오류, 타이밍 미스
- 조치: 디버그 모드에서 히트박스 시각화, 서버 피격 로그 확인

- 세션 및 룸 관리 문제

- 문제: 유령 유저 잔류, 룸 중복 참가
- 원인: 입퇴장 처리 미흡, 룸 생성/해제 타이밍 문제
- 조치: 입퇴장 로그 기록, 유휴 세션 자동 삭제, 동시 접속 제한 강화

- 입력 처리 오류

- 문제: 키 입력 누락, 터치 입력 미반영
- 원인: 이벤트 중복 바인딩, 해제 누락, 뷰포트 해상도 차이
- 조치: 입력 로그 출력, 버퍼 및 디바운스 적용, 터치 입력 전용 인터페이스 활성화

- 애니메이션 및 UI 불일치

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

- 문제: 공격 모션과 판정 타이밍 불일치, HUD 업데이트 지연
- 원인: 애니메이션 재생과 로직 처리 분리 미흡, 상태 이벤트 누락
- 조치: 로직과 UI 트리거 분리, 상태 갱신 이벤트 로그 모니터링

향후 확장 계획

- 게임 모드 확장 구조 (팀전, 레이드, 점프맵 등 설계 방향)
- 사용자 성장 시스템 구상 (경험치, 무기 숙련도 등)
- 캐릭터/무기 커스터마이징 UI 도입 방향
- 로그인, 랭킹, 채팅 시스템과의 연계 가능성
- 모바일 최적화 및 반응형 대응 계획

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

4. 개별 후기

● 이영호 후기

전투 시스템 구현을 주도하며, 근접 무기부터 원거리 무기에 이르기까지 다양한 무기별 공격 판정과 피격 처리, 시각적 피드백 효과를 설계하고 개발했습니다. 초기 단계에서 공격 판정 로직이 지나치게 복잡하고 중복되는 부분이 많아, 코드의 비효율성과 버그 발생 위험이 높았습니다. 특히, 서버와 클라이언트 간 판정 불일치 현상으로 인해 전투 중 부정확한 피해 판정 사례가 다수 발생해 많은 디버깅 시간이 소요되었습니다. 하지만 반복적인 테스트와 로깅, 코드 리팩토링 과정을 통해 핵심 로직을 단순화하고 명확한 판정 기준을 세워 문제를 극복했습니다.

또한, 애니메이션 시스템과 공격 로직을 분리하여 게임플레이가 보다 자연스럽게 직관적으로 느껴지도록 개선한 점은 큰 성과였습니다. 다만, 프로젝트 일정과 리소스 제한으로 인해 무기별 밸런스 조정에 충분한 시간을 할애하지 못해, 완성도 면에서 다소 아쉬움이 남습니다. 향후에는 더욱 체계적인 밸런스 테스트 및 자동화 도구 도입을 계획하고 있습니다.

● 김기찬 후기

서버 구축 및 네트워크 통신 안정화에 집중하여, Node.js 기반의 실시간 게임 서버를 설계 및 운영했습니다. Socket.IO를 활용한 실시간 이벤트 기반 통신 체계를 구현하는 과정에서, 네트워크 지연 및 패킷 손실 문제로 인한 데이터 동기화 오류가 빈번히 발생했습니다. 특히, 클라이언트와 서버 간 위치 불일치 및 공격 판정 차이로 인해 게임 경험에 영향을 주는 문제가 컸습니다. 이에 ping 측정과 시간 보정 알고리즘을 도입하고, 클라이언트 측 보간(LERP) 방식을 개선하는 등 다각도로 접근해 동기화 문제를 완화했습니다.

또한, 서버 자원의 효율적 관리를 위해 유휴 세션 자동 삭제 기능을 구현하고, 예외 케이스 처리 로직을 강화하여 시스템 안정성을 크게 높였습니다. 다만, 부하 테스트가 충분히 진행되지 않아 다수 동시 접속 환경에서의 확장성 검증이 미흡했으며, 이에 대한 보완 작업이 향후 과제로 남았습니다. 장기적으로는 Redis Pub/Sub나 Socket.IO 어댑터 기반의 수평 확장 전략도 준비 중입니다.

● 조유빈 후기

UI 설계와 3D 맵 렌더링 구현을 담당하며, Three.js를 활용해 다양한 맵 테마와 캐릭터 선택 인터페이스를 구성했습니다. 초기 개발 단계에서 데스크탑 환경 중심으로 설계한 UI가 모바일 환경에서는 터치 입력 오류와 반응 속도 저하 문제를 보였으며, 이는 사용자 경험에 직접적인 악영향을 끼쳤습니다. 터치 영역 불일치, 키 입력 중복 발생 등 다양한 문제를 확인했고, 입력 버퍼링과 디바운스 로직 적용, 모바일 환경

프로젝트 완료 보고서		
프로젝트 주제	Node.js 서버를 기반으로 한 3D 웹 게임 개발	
단계 : 프로젝트 완료	작성자 : 이영호	작성일 : 2025.07.28

에 특화된 인터페이스 감지 기능을 추가하며 문제를 완화시켰습니다.

또한, 네온 효과와 모달 팝업 등 다양한 시각적 효과를 구현하는 과정에서 코드 복잡도가 증가했으나, 지속적인 코드 리팩토링과 모듈화 작업을 통해 유지보수성을 개선했습니다. 반응형 디자인 구현 시 화면 비율과 해상도 차이를 충분히 고려하지 못해 발생한 일부 UI 왜곡 문제도 있었으나, 테스트와 피드백을 거쳐 상당 부분 수정 완료했습니다. 앞으로는 다양한 디바이스 및 브라우저 환경에 대한 체계적인 테스트 프로세스를 마련해 사용자 접근성을 높이하고자 합니다.