요구사항 분석 & 프로젝트 설계서 3차

텍스트, 엠블럼, 로고, 원이(가) 표시된 사진

자동 생성된 설명]

**팀: 컴공이면 게임해야조**

**제출 일시: 2024 - 11 - 08**

목차

[**1. 개요**](#_e9b76bxdlr6g)

[1.1. 기획 배경](#_lllznwas7a12)

[1.2. 괸련 동향](#_3tmyhn2ws4qa)

[1.3. 프로젝트 주요 기능 및 특징](#_mcjg3dr2svfl)

[1.3.1. 서비스의 흥미 요소](#_u7n6quedi70t)

[1.3.2. 목표](#_oiwndea9ggxc)

[1.4. 조원 구성 및 역할 분담](#_us5au9fdp2z3)

[1.5. 일정](#_us5au9fdp2z3)

[**2. 기능적 요구 사항**](#_rf0rx98vpssx)

[2.1. Top Level Usecase Diagram](#_ey00jdydpibf)

[2.2. 기능별 동작 시나리오](#_4mmvtnji7jf0)

[**3. 비기능적 요구사항**](#_7i4sjfgls56r)

[3.1. 사용 편리성](#_2t73olui2qjf)

[3.2. 신뢰성](#_9bpjmxgvfj26)

[3.3. 성능](#_zg08o987hajs)

[3.4. 구현](#_9bpjmxgvfj26)

[3.5. 인터페이스](#_zg08o987hajs)

[**4. 소프트웨어 Top-level 구조**](#_7i4sjfgls56r)

[**5. 컴포넌트별 기능 정의**](#_7i4sjfgls56r)

[**6. 컴포넌트 – Server 설계**](#_7i4sjfgls56r)

[**7. 컴포넌트 – VR 설계**](#_7i4sjfgls56r)

[**8. 컴포넌트 – Web 설계**](#_7i4sjfgls56r)

[**9. 주요 기능 실행 화면**](#_7i4sjfgls56r)

[**10. 시스템 통합 시험**](#_7i4sjfgls56r)

[10.1. 기능 시험 결과](#_9bpjmxgvfj26)

[10.2. 비기능 시험 결과](#_zg08o987hajs)

[**11. 결론**](#_7i4sjfgls56r)

[**<부록1> 사용자 설명서**](#_7i4sjfgls56r)

[**<부록2> 컴파일 및 설치**](#_7i4sjfgls56r)

[**<부록3> 소스 코드**](#_7i4sjfgls56r)

1. 개요

1.1 기획 배경

체스와 게임에 흥미 있는 학우들끼리 모여 프로젝트에 관해 생각하던 중, 다른 사람들도 체스의 재미를 느낄 수 있도록, 무거운 체스의 분위기를 덜어낸 게임을 만들고자 하였습니다. 이를 위해서 2:2 팀 게임 방식을 도입하였으며, VR 기기를 추가하여 입문자도 쉽게 즐길 수 있는 게임을 만들고자 하였습니다.

기존 서비스와의 차별점

1. 몰입형 VR 체스 경험

* 현실감: 기존의 2D 온라인 체스와 달리, VR 기기를 통해 사용자가 실제 기물이 되는 경험을 제공합니다.
* 고해상도 그래픽: 최신 VR 기술을 활용한 고해상도 그래픽과 3D 환경으로, 게임의 몰입감을 극대화합니다.

2. 2:2 팀 모드

* 팀 기반 게임플레이: 전통적인 1:1 체스와 달리, 2:2 팀 모드를 도입하여 협력과 전략적 협동의 재미를 제공합니다. 팀원들과 함께 전략을 세우고 협력하여 게임을 진행할 수 있습니다.
* 다양한 전략적 요소: 팀 게임은 아이템 등 더 다양한 전략적 요소를 도입할 수 있어 게임의 깊이와 재미를 더합니다.

3. 멀티플레이어 및 소셜 기능

* 실시간 소셜 인터랙션: 다른 사용자와 실시간으로 소통하고, 친구와 팀을 이루는 경험을 제공합니다. 이는 기존 온라인 체스 게임의 제한된 상호작용을 넘어서는 기능입니다.

5. 공포 테마와 환경

* 커스터마이즈 가능: 공포 분위기의 체스판 테마와 배경을 제공하여, 게임의 분위기를 긴장감 있게 만듭니다. 이는 기존 체스 게임에서 제공하지 않는 시각적 다양성을 추가합니다.
* 역동적인 환경: 고정된 체스판이 아닌 다양한 환경에서 체스를 즐길 수 있어, 매번 새로운 경험을 제공합니다.

**결론**

VR 체스 게임은 이러한 기존 서비스의 한계를 극복하고, 체스의 재미와 교육적 가치를 극대화할 수 있는 잠재력을 가지고 있습니다. 이를 통해 더 많은 사람들이 체스를 즐기고 배울 수 있으며, 몰입감 있는 소셜 게임 경험을 제공할 수 있습니다.

1.2 관련 동향

* **몰입형 경험의 증대**:

**고해상도 디스플레이**: VR 기기의 해상도와 화질이 계속해서 개선되고 있습니다. 예를 들어, Meta Quest 3와 같은 최신 VR 헤드셋은 뛰어난 해상도를 제공하여 사용자의 몰입감을 극대화합니다.

**트래킹 기술의 발전**: 몸 전체를 추적하는 트래킹 기술이 발전하면서 게임 내 캐릭터의 움직임이 더욱 자연스럽게 표현되고 있습니다. 실제로 움직이는 듯한 현실감 있는 체험을 제공합니다.

* **소셜 인터랙션 강화**:

**멀티플레이어 기능**: VR 게임에서 다른 사용자와 함께 게임을 즐길 수 있는 멀티플레이어 기능이 중요해지고 있습니다. 이는 친구나 전 세계의 플레이어와 함께 게임을 즐길 수 있는 기회를 제공합니다.

**가상 커뮤니티**: VRChat과 같은 플랫폼은 사용자가 가상 공간에서 다른 사람들과 소통할 수 있는 기능을 제공하여, 게임 외에도 다양한 소셜 활동을 즐길 수 있습니다.

* **접근성 향상**:

**직관적인 인터페이스**: VR 기기의 인터페이스가 점점 직관적으로 변하면서, 비게임 유저나 기술에 익숙하지 않은 사람들도 쉽게 접근할 수 있게 되었습니다.

**경량화 및 무선화**: VR 기기가 점점 가벼워지고 무선으로 변하면서 사용자의 이동성을 높이고, 장시간 사용해도 피로감을 줄여줍니다.

1.3 프로젝트 주요 기능 및 특징

2:2 팀 게임으로 진행됩니다. A팀과 B팀이 있다고 했을 때 A팀과 B팀에는 각각 명령자와 수행자가 존재합니다. 명령자는 Web을 통해 온라인 체스를 플레이하는 것과 같은 방식으로 명령을 내립니다. 수행자는 VR 기기를 착용하며, 명령자에게 명령을 받으면 직접 기물이 되어 움직이고 전투함으로써 명령을 수행합니다.

A팀의 명령자가 기물을 움직이면 서버에 전달되고, 움직임 정보는 A팀의 행동자에게 전달됩니다. A팀의 행동자는 명령을 받으면 기물이 되어서 움직입니다. 이후 B팀의 명령자와 행동자도 동일한 방식으로 차례를 진행합니다.

요약하면, A팀 명령자 움직임 -> A팀 행동자에게 전달 -> A팀 행동자 움직임 -> B팀 명령자 움직임 -> B팀 행동자에게 전달 -> B팀 행동자 움직임 게임이 끝날 때까지 이 과정을 반복합니다.

1.3.1 서비스의 흥미 요소

VR 사용자에게는 체스판 위의 공포 분위기를 통해 긴장감을 유지시키며, 이동과 전투, 아이템 사용 등 다양한 사용자 경험을 제공합니다.

Web 사용자는 체스 자체의 즐거움뿐만 아니라, 마치 공포게임 방송을 보는 것처럼VR 사용자의 체험을 관전하는 재미를 제공합니다.

팀 플레이를 통해 서로 간의 친밀감과 유대감 형성에 기여합니다.

1.3.2 목표

Web과 VR 사이 원활한 플레이가 가능하도록 커뮤니케이션 딜레이를 줄입니다.

VR 사용자의 경우 움직임에 이질감이 들지 않는 사용자 경험을 제공합니다.

1.4 조원 구성 및 역할 분담

- 프론트 엔드 팀

정성록: 기획, 모션 및 클라이언트 설계, 인 게임 모션 구현

서지우: 기획, 모션 및 클라이언트 설계, 인 게임 모션 구현

- 백 엔드 팀

문무현: 기획, 체스 로직 및 아이템 기능 구현, 서버 및 Web 클라이언트 구현

전성윤: 기획, 체스 로직 및 아이템 기능 구현, 서버 및 Web 클라이언트 구현

1.5 일정

**현재 진행 상황**

Server – 동작에 필요한 api가 정의되어 있으며 Web과 VR 각각의 클라이언트의 요청을 처리할 수 있음.

VR – 전체 보드와 기물별 맵이 구현되어 있으며 이동과 전투 이벤트를 실행할 수 있음.

Web – 화면에 보드가 표시되며 상호작용을 통해 기물을 동작할 수 있음.

**향후 일정**

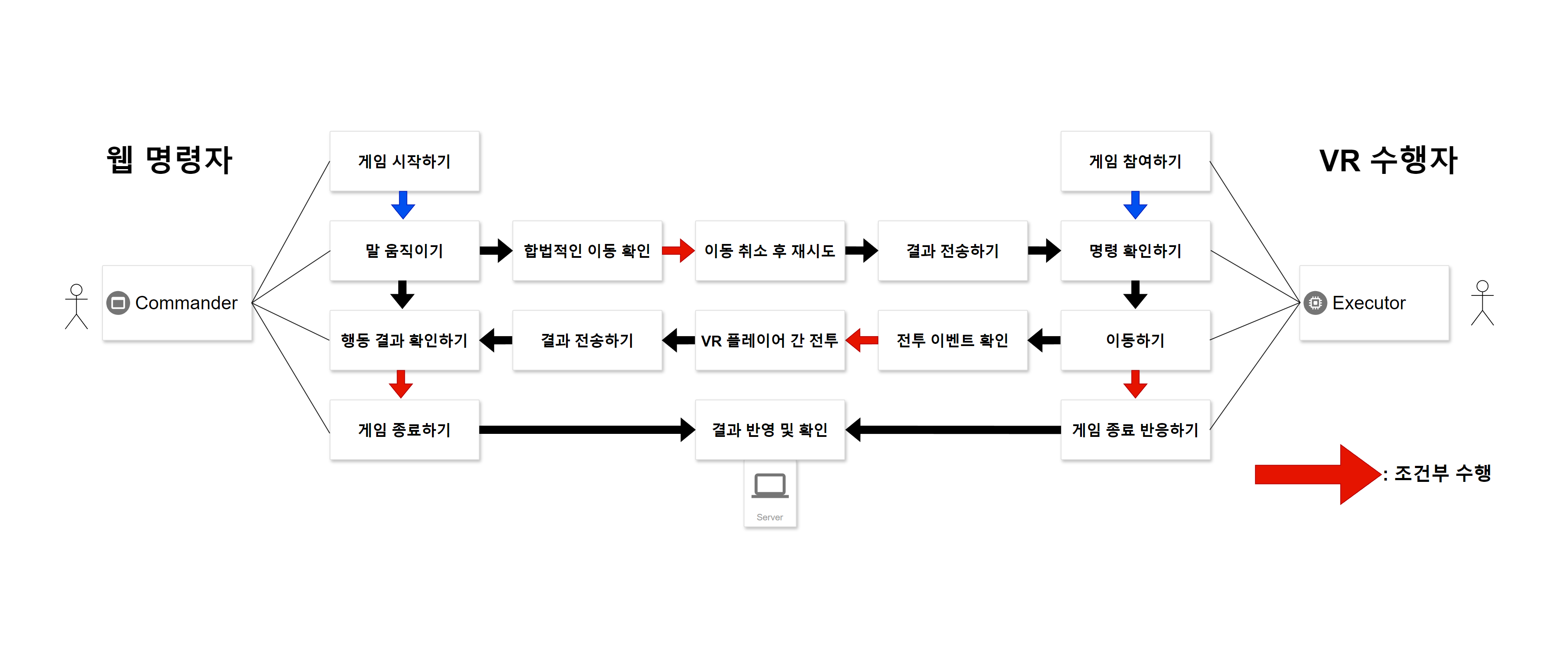
~10/27: Web과 VR 클라이언트를 서버를 통해 연결한다.

~11/17: 기능/비기능 테스트를 진행한다.

~12/7: 오류 수정 및 발표, 보고서를 작성한다.

2. 기능적 요구사항

2.1 Use Case Diagram



한 팀 내에서 명령자와 행동자의 Usecase를 나타내는 다이어그램입니다.

1.4 서비스의 소개를 통해 게임의 진행 방식을 참고바랍니다.

2.2 각 기능별 동작 시나리오

- 이동

|  |  |
| --- | --- |
| 기물 이동 | |
| Name of Use Case | 기물 이동 |
| Actors | 1P(명령자), 2P |
| Flow of Events | 1. 1P : 이동명령 , 2P : 이동명령 확인  2. 2P : 이동  3. 2P : 도착 신호 보냄  4. 위치 전송  5. 위치정보 확인  6. 턴 종료 |
| Exceptions | 5에서 상대 기물 확인되면 전투  5에서 갈 수 있는 위치에서 벗어났다면 2번으로 회귀 |

- 전투

|  |  |
| --- | --- |
| 전투 기능 | |
| Name of Use Case | 기물 이동 |
| Actors | 2P(수행자) |
| Flow of Events | 1. 상대 기물에 나타난 선 확인  2. 선을 따라 공격  3. 전투 스코어 및 시간 서버에 전송  4. 공격 성공/실패 판정 전송  5. 상대 기물 파괴 후 칸 차지 |
| Exceptions | 4에서 공격실패 판정 시 본인이 파괴됨 |

3. 비기능적 요구사항

3.1 사용 편리성

Web 사용자에게 게임 진행 상황을 조금 더 직관적으로 살펴보며 플레이할 수 있도록 인터페이스 구성합니다.

VR 사용자에게 게임 내 움직임을 적응할 수 있도록 VR 기기 체화를 위한 튜토리얼을 제공합니다.

기존 체스게임과의 새로운 규칙을 설명하기 위한 설명서 제공합니다.

3.2 신뢰성

플레이에 불편함이 없도록 99.999%의 가용성 보장합니다.

그래픽 깨짐 현상으로 인한 멀미 유발 방지를 위한 그래픽 퍼포먼스 최적화합니다.

3.3 성능

턴 제 게임이기 때문에 불편하지 않을 정도의 지연시간 (30~40ms) 유지합니다.

VR 사용자 간 전투 시 실시간으로 진행되는 듯한 느낌을 받도록 100ms 응답시간으로 전투 실행 결과 확인합니다.

3.4 구현

VR Client - Unreal Engine 5 (Epic Game Store 에셋 활용)

Web Client – React

Server – Spring boot

3.5 인터페이스

RESTful API 사용하여 클라이언트와 서버 연결합니다.

4. 소프트웨어 Top-level 구조

4.1 Component Diagram

스크린샷, 도표, 텍스트이(가) 표시된 사진

자동 생성된 설명

5. 컴포넌트별 기능 정의

- **Component Diagram** 설명

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

**▶User :**

VR 사용자와 웹 사용자

각각의 인터페이스를 통해 상호작용

**▶Web Socket :**

VR 및 웹 인터페이스 모두 서버와의 통신에 웹 소켓 연결

텍스트, 스크린샷, 도표이(가) 표시된 사진

자동 생성된 설명

**▶Move :**

입력 : 이동 검사 결과, 아이템 사용 여부

출력 : 전투 발생 판별, 이동 후 기물 좌표

**▶Battle :**

입력 : 이동 검사 결과, 아이템 사용 여부

출력 : 전투 결과

**▶Time & Turn & Item :**

입력 : 웹 및 VR 클라이언트 명령

출력 : 웹 및 VR 클라이언트의 턴, 남은 시간 및 아이템

**▶Board Data :**

입력 : 이동 및 전투 결과

출력 : 현 보드 상태

**▶Players Data :**

입력 : 남은 시간, 턴 및 사용한 아이템

출력 : 시간, 턴 및 아이템 갱신 결과

**▶State Update :**

입력 : 보드와 플레이어 데이터

출력 : 보드와 플레이어 데이터

**▶Result Data Process :**

입력 : 보드와 플레이어 데이터

출력 : 클라이언트에 전송하기 위해 분류 및 패키징 한 메시지

6. 컴포넌트 모듈 설계 – Server

6.1 컴포넌트 모듈 클래스 구성도

WebSocketConfig (웹소켓 설정)

* 브로커 설정 (구독 및 메시지 전송).
* STOMP 프로토콜을 통해 클라이언트와 서버 간의 통신을 설정.

ChessController (실시간 체스 게임 관리)

* 클라이언트 간의 체스 게임 상태 관리.
* 웹과 VR 클라이언트 참여 관리.
* 체스 말 이동 처리.
* STOMP 메시징 템플릿을 사용하여 메시지 전송

Move (체스 말의 이동)

* 역할: 체스 말이 움직인 정보를 관리합니다.

Position (체스판의 위치)

* 역할: 체스판의 위치 좌표를 나타내는 클래스입니다.

ChessPiece (체스 말)

* 역할: 각 체스 말의 타입, 색상, 위치를 관리합니다.
* gamestate (게임 상태 저장 및c 관리)
* controller로부터 전달받은 사용자의 요청을 처리하고 상태를 저장.

6.2 각 클래스들의 설계

6.2.1 멤버 데이터 설명

WebSocketConfig 주요 데이터:

없음.

ChessController 주요 데이터:

gameState: 게임 상태를 관리하는 서비스 클래스 객체

connectedWebSessions: 현재 접속한 웹 사용자 세션을 관리

connectedVRSessions: 현재 접속한 VR 사용자 세션을 관리

Move (체스 말의 이동)

eventTime: 체스 말이 움직인 시간.

from: 말이 움직이기 전의 위치 (Position 객체).

to: 말이 이동한 후의 위치 (Position 객체).

color: 말의 색상 (흑 또는 백).

type: 말의 종류 (룩, 비숍 등).

Position (체스판의 위치)

x: 체스판의 가로 좌표.

y: 체스판의 세로 좌표.

ChessPiece (체스 말)

type: 말의 종류 (룩, 비숍 등)

color: 말의 색상 (흑/백)

position: 말의 현재 위치 (Position 객체)

board  
currentPlayer: 현재 플레이어의 차례를 나타냄(White or Black)  
currentRole: 현재 플레이어의 역할을 나타냄(Commander or Actor)  
whiteKingMoved, blackKingMoved: 캐슬링 가능 여부 판별을 위한 변수  
whiteRook1Moved, whiteRook2Moved: 캐슬링 가능 여부 판별을 위한 변수  
blackRook1Moved, blackRook2Moved: 캐슬링 가능 여부 판별을 위한 변수  
enPassantTarget: 앙파상 처리를 위한 변수

6.2.2 멤버 함수 설명

WebSocketConfig 주요 메서드:

* configureMessageBroker(MessageBrokerRegistry config): 메시지 브로커 설정.
* registerStompEndpoints(StompEndpointRegistry registry): STOMP 엔드포인트 등록.

ChessController 주요 메서드:

Webjoin(SimpMessageHeaderAccessor headerAccessor): 웹 사용자가 게임에 참여하는 기능.

VRjoin(SimpMessageHeaderAccessor headerAccessor): VR 사용자가 게임에 참여하는 기능.

 processMoveWeb(String move, StompHeaderAccessor headerAccessor): 웹 사용자의 체스 말 움직임을 처리.

 processMoveVR(String move, StompHeaderAccessor headerAccessor): VR 사용자의 체스 말 움직임을 처리.

 handleTimeUp(String message): 웹 사용자가 제한 시간을 초과한 경우 처리

 resetGame(): 게임을 재시작하는 경우 처리

Move (체스 말의 이동)

Move(String eventTime, Position from, Position to, String color, String type): 체스 말의 이동을 생성합니다.

Move(Position from, Position to): 기본 이동 생성자.

Position (체스판의 위치)

fromString(String positionString): 문자열을 받아 위치 좌표를 생성하는 정적 메서드.

equals(): 두 위치가 같은지 비교하는 메서드.

hashCode(): 위치 좌표의 해시코드를 생성.

ChessPiece (체스 말)

equals(): 체스 말의 위치가 같은지 비교.

hashCode(): 위치에 따른 해시코드를 생성.

initializeBoard(): 보드 상태 초기화

processMoveWEB(Move move), processMoveVR(Move move): move 이벤트 처리

getBoardState(): 현재 보드 상태 출력

updateBoard(Move move): 보드 상태 업데이트

switchPlayer(), switchTurn(): 색상 및 역할 변경

isValidMove(Move move, Map<Position, ChessPiece> board): 유효한 움직임인지 확인

isPathClear(Position from, Position to, Map<Position, ChessPiece> board): 경로 상에 장애물이 있는지 확인

isCheckmate(): 체크메이트인지 확인

handleSpecialMoves(Move move, ChessPiece piece): 앙파상, 캐슬링, 프로모션 처리

7. 컴포넌트 모듈 설계 - VR

7.1 컴포넌트 모듈 클래스 구성도

EnemyCharacter\_BP (Rampage & Khaimera & Narbash):

적 AI 행동 제어 (플레이어 추적, 공격).

Perception 및 반응 시스템을 포함하여 플레이어를 감지.

TriggerBox\_BP:

플레이어가 특정 영역에 들어가면 이벤트를 트리거하는 컴포넌트 (예: 적 출현, 불빛/음향 조절).

AIController\_BP:

적 캐릭터의 순찰 및 플레이어 감지 시 반응하는 AI 컨트롤러.

VRPawn\_BP:

- 플레이어의 메인 캐릭터.

- 플레이어의 시점, 카메라, 그리고 모션 입력을 처리하는 VR 컨트롤러.

7.2 각 클래스들의 설계

7.2. 클래스 설명

VRPawn\_BP:

-VRPawn: 플레이어를 관리하는 VR 컨트롤러.

-MovementInput: 플레이어 움직임 입력을 처리 (업로드한 player\_move 블루프린트 참고).

EnemyCharacter\_BP (Rampage & Khaimera):

-Player: 적이 상호작용하는 플레이어의 참조.

-AI\_Perception: 감지 논리를 처리하여 (시야) 추적 행동을 트리거.

-IsVisible: 적이 숨겨져 있는지, 보이는지를 결정하는 Boolean (업로드한 triggerbox 블루프린트 참고).

TriggerBox\_BP:

-Player: 오버랩 이벤트를 트리거하는 플레이어 캐릭터에 대한 참조.

-IsVisible: 적이 나타나고 플레이어를 추적하기 시작할지 여부를 결정.

AIController\_BP:

-PathFollowingComponent: AI의 경로를 제어하는 컴포넌트.

-DetectedActor: 적이 플레이어를 감지했을 때 참조하는 액터.

8. 컴포넌트 모듈 설계 - WEB

8.1 컴포넌트 모듈 클래스 구성도

- App.js

: 양방향 통신이 가능한 stomp메시지로 서버와 데이터를 주고 받는 메인 컴포넌트

- Board.js

: 보드 상태를 표현하는 컴포넌트

- Piece.js

: 각 기물의 이미지, 위치, 색상 등을 나타내는 컴포넌트

- Square.js

: 보드의 사각형 판의 위치, 색상 등을 나타내는 컴포넌트

8.2 각 클래스들의 설계

8.2.1 멤버 데이터 설명

- App.js

selectTeam: 팀 선택 변수

initialSeconds: 초기 시간 변수

seconds: 현재 시간 변수

timeOwner: 현재 턴인 색상

isReversed: 상하 반전 여부

gameStarted: 게임 시작 여부

invalidMoveMessage: 유효하지 않은 기물을 움직일 때 보낼 메시지

invalidMoveFlag: 기물 움직임이 잘못됨 나타내는 플래그

validMoveFlag: 기물 움직임이 유효함을 나타내는 플래그

resetBoardFlag: 보드를 리셋하는데 사용할 변수

boardState: 보드 상태 정보를 받을 변수

- Board.js

board: 현재 보드 상태를 나타냄

draggedPiece: 마우스로 드래그한 기물 정보 저장

draggedOverSquare: 사용자가 드래그한 기물을 놓을(드롭할) 위치를 저장

prevBoard: 이전 보드 상태 저장

- Piece.js

pieceImage: 색상, 종류에 따라 기물 이미지를 저장하는 변수

- Square.js

black: 보드의 검은색 사각형 판의 위치

className: 보드의 사각형 판의 이름

8.2.2 멤버 함수 설명

- App.js

connect: STOMP 프로토콜을 사용하여 웹소켓 서버에 연결

subscribe: STOMP 클라이언트를 사용하여 서버로부터 메시지를 구독

disConnect: STOMP 클라이언트를 사용하여 현재 웹소켓 연결을 끊기

msg\_callback: broker 가 client 한테 메시지 전송할때마다, 트리거되는 콜백 함수

sendMoveData: web 정보를 서버로 전달

sendTimeUpMessage: seconds가 0이 되었을 때 서버로 메시지를 보내는 함수

resetMessage: reset버튼 눌렀을 때 서버에 메시지를 보내는 함수

completeChooseTeam: 팀 선택 후 결과를 서버로 보냄

- Board.js

handleDragStart: 기물을 드래그하기 시작할 때 호출, 드래그 시작 시 현재 보드 상태(prevBoard.current)를 저장하고, 드래그한 기물의 정보(draggedPiece)와 위치(position)를 상태로 저장

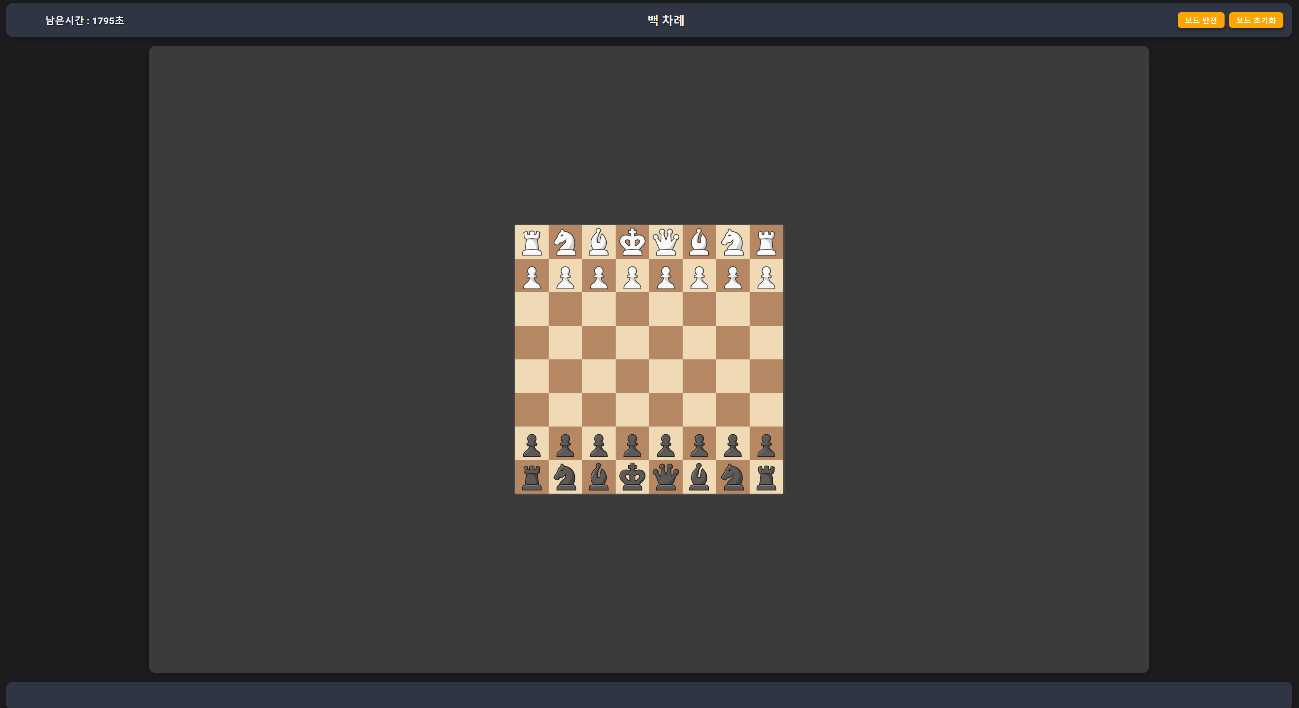
handleDragOver: 사용자가 기물을 드래그 중인 칸 위로 마우스를 올릴 때 호출, e.preventDefault()를 사용해 기본 드래그 이벤트를 방지하고, 드래그 오버한 칸의 위치를 draggedOverSquare로 저장

handleDrop: 기물을 드래그 후 놓을 때 호출, 기물의 움직임 정보를 서버로 전송하며, 움직임이 완료된 후 상태를 초기화

renderSquare: 보드의 각 칸을 생성하는 함수

9. 주요 기능 실행 화면

웹 브라우저 실행 화면



VR 화면

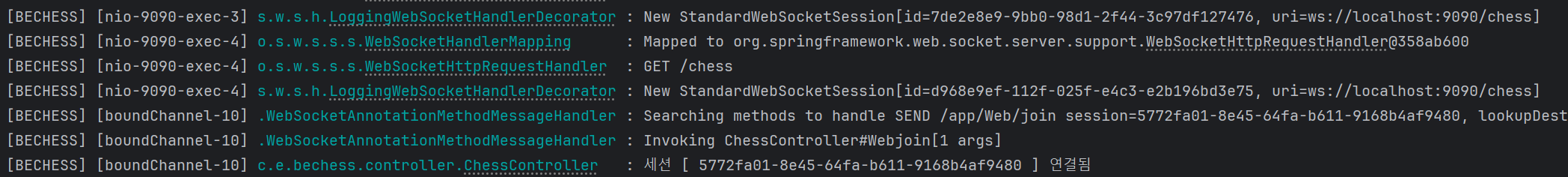
만화 영화, 댄스, 장난감, 스크린샷이(가) 표시된 사진

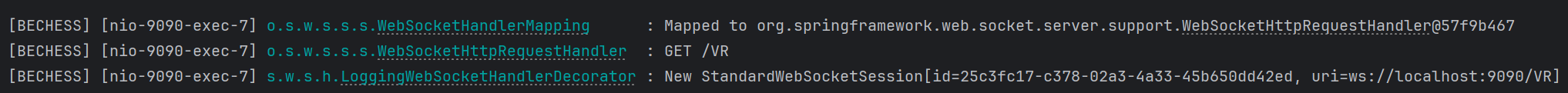
자동 생성된 설명

어둠, 야외, 안개, 밤이(가) 표시된 사진

자동 생성된 설명

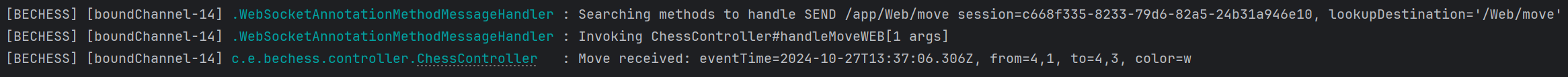
서버 실행 화면





스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명



10. 시스템 통합 시험

10.1 기능 시험 결과

|  |  |  |
| --- | --- | --- |
| 기능 명세서 | 결과 | 사진 |
| 게임 실행 | 성공 |  |
| 이동 처리 | 성공 |  |
| 전투 처리 | 대기 |  |
| 순서 처리 |  |  |
| 실시간 메시징 | 성공 |  |
| 종료 및 재실행 | 성공 |  |

10.2 비기능 시험 결과

|  |  |  |
| --- | --- | --- |
| 비기능 명세서 | 결과 | 사진 |
| 보안 테스트 | 성공 |  |
| 사용성 테스트 | 대기 |  |
| 가용성 테스트 | 성공 |  |
| 확장성 | 성공 |  |
| 유지보수성 | 대기 |  |
| 안정성 | 성공 |  |

11. 결론

실시간 상호 작용을 통해 VR과 Web에서 동일한 게임 상황을 공유하는 2:2 대전 게임을 개발하였습니다. VR을 도입하여 게임의 몰입도와, 2:2 플레이를 통한 소셜 경험을 제공하여 체스라는 보드게임에 전에 없던 새로운 재미를 부여합니다.

현재 네 명의 플레이어만을 지원하지만 데이터베이스 확장을 통해 개별 플레이어와 로비 기능 등을 지원하는 방식으로 게임을 확장할 수 있습니다.

<부록 1> 사용자설명서

2:2 체스 게임으로 WEB 기반 명령자와 VR 기반 수행자가 한 팀이 되어 다른 팀을 상대합니다.

명령자가 브라우저에서 기물을 움직여 명령을 내리면 수행자는 VR 환경 내에서 1인칭 시점으로 직접 그 기물이 되어 이동과 전투를 수행합니다. 수행자는 전체 보드 상황을 파악하기 어렵습니다.

수행자는 이동이 체스 규칙에 벗어나지 않는 경우 명령자의 명령과 다른 곳으로 이동할 수 있습니다. 전투에서 패배한 경우 자기 기물이 파괴됩니다.

게임은 WEB white의 명령 -> VR white의 수행 -> Web black의 명령 -> VR black의 수행 -> Web white의 명령이 순서대로 게임이 종료될 때까지 반복됩니다.

이외 모든 게임 규칙은 일반적인 체스의 규칙을 따릅니다.

<부록 2> 컴파일 및 설치

최소 웹 브라우저가 실행 가능한 컴퓨터 2대와 VR 기기 2대가 필요합니다. WEB 클라이언트 컴퓨터 2대와 서버 실행 컴퓨터 1대를 별도로 준비하는 것을 권장합니다. 모든 기기는 네트워크에 접속 가능해야 하며 컴퓨터가 2대인 경우 한 컴퓨터에서 서버와 WEB 클라이언트의 역할을 모두 수행해야 합니다.

<https://github.com/MoonMoohyeon/BECHESS>

<https://github.com/MoonMoohyeon/BECHESS-WEB>

다음 주소에서 소스코드를 다운로드 받아서 실행합니다. React, springboot 등 별도 실행 환경을 준비해야 합니다.

WEB 클라이언트 파일에서 App.js의 brokerURL을 서버의 호스트 주소로 변경한 후 실행합니다. 포트 번호 변경이 필요한 경우 서버 코드의 application.properties의 server.port를 원하는 포트 번호로 변경할 수 있습니다.

<부록 3> 소스코드

<https://github.com/MoonMoohyeon/BECHESS>

<https://github.com/MoonMoohyeon/BECHESS-WEB>