PRESENTATION

2025.07.02

실시간 웹 기술과 AI를 결합한 **오목 대전 서비스**

2403110265 김해준

한국폴리텍대학 성남캠퍼스
인공지능소프트웨어과

CONTENTS

목차

서비스 개요 및 개발 목적

- **문제점**

  기존 오목 게임은
  - 단순한 PvP 기능에 국한
  - AI 수준이 낮아 예측 가능
  - 소셜/인증 기능 부족

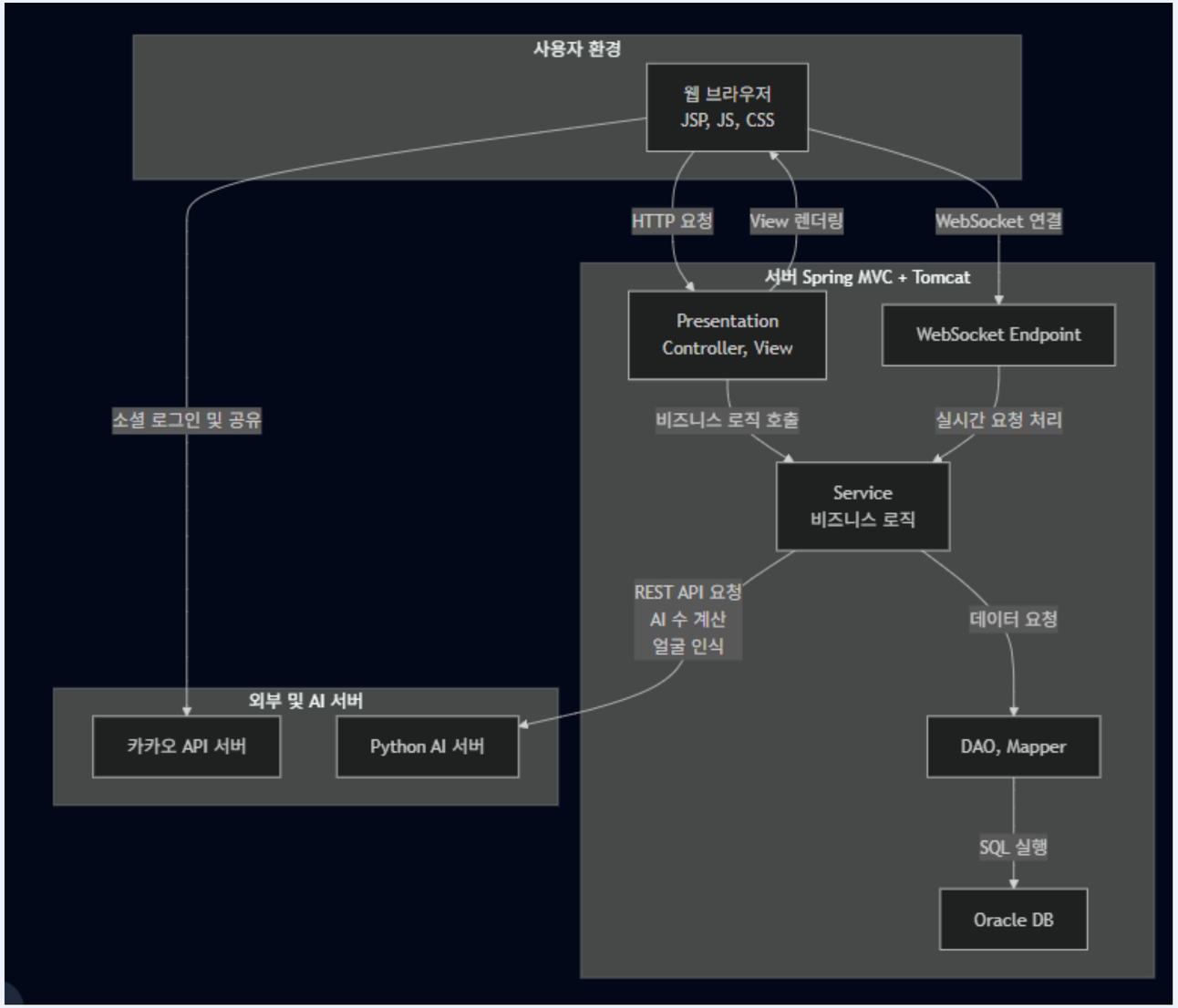
- **프로젝트 목적**

  끊김 없는 실시간 대전 환경 제공
  강화학습 AI와의 대결
  카카오 소셜 로그인, 얼굴 인식 로그인 등의 기술을 통해
  사용자 편의성 향상

주요 기능 요약

구분	주요기능
대전	실시간 PvP, 강화학습 AI 대전(PvE), 게임 관전, 복기
사용자	일반/소셜/게스트/얼굴인식 로그인, 마이페이지, 프로필 수정/탈퇴
소셜/기타	실시간 채팅(욕설 필터링), 카카오톡 초대, 사용자 신고
관리자	회원 관리(검색, 정지), 방 관리(좀비방 삭제), 신고 내역 조회

전체 시스템 구성도



1 클라이언트

- JSP와 CSS를 기반으로 사용자 인터페이스 제공
- 웹소켓을 이용해 서버와 실시간으로 데이터를 교환하며 동적인 상호작용

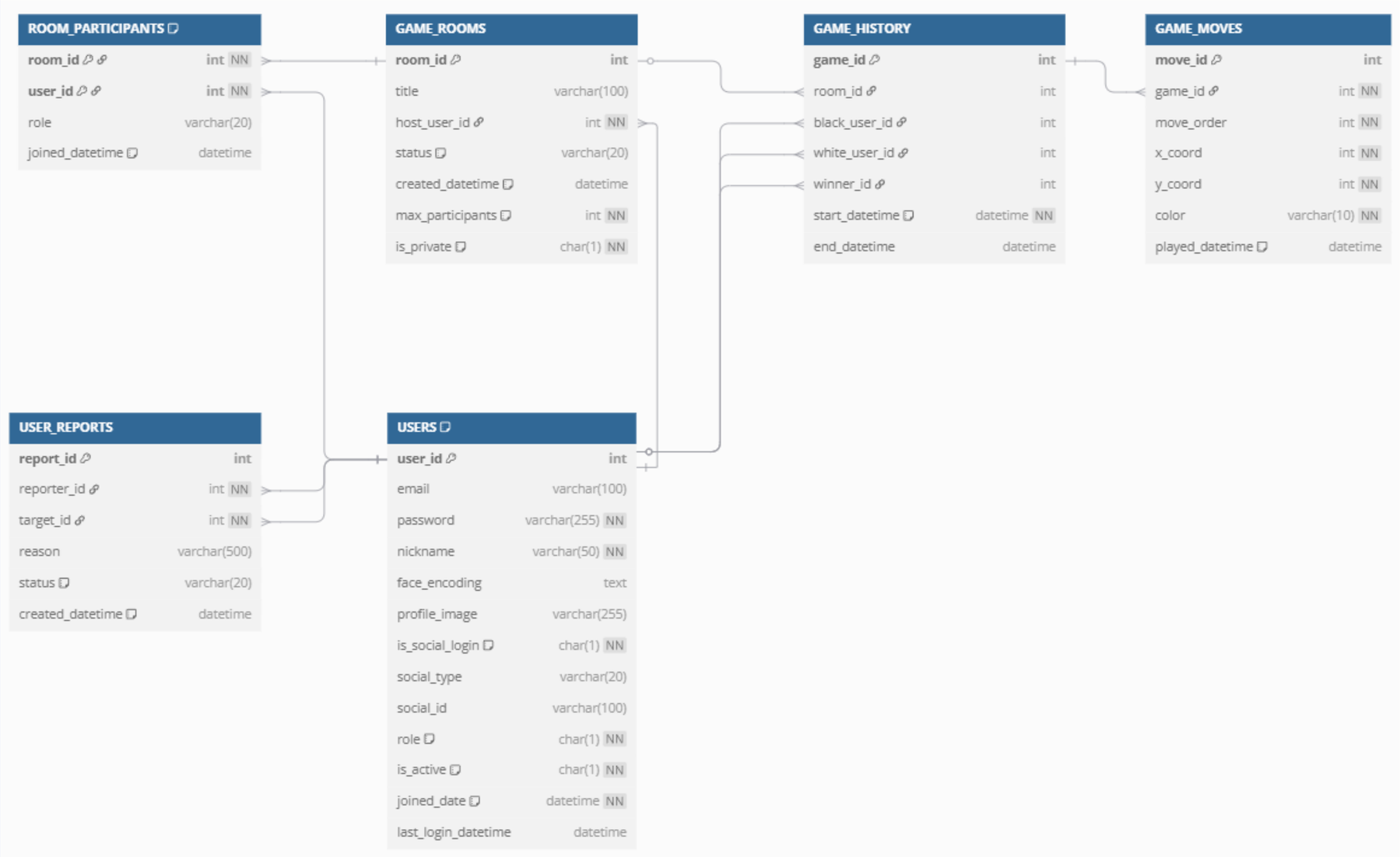
2 웹 애플리케이션 서버 및 데이터베이스

- Spring MVC 프레임워크 기반의 서버
- 웹소켓 엔드포인트가 실시간 게임 상태와 사용자 연결
- Oracle 데이터베이스와 연동

3 외부 연동 서버

- 카카오API 서버와 통신하여 사용자 인증 및 공유 기능
- Python AI 서버를 별도로 구축하여, 강화학습으로 훈련된 AI 모델 사용
- Spring 서버는 REST API를 통해 Python 서버에 다음 수를 요청하고 응답받아 AI 대전을 구현

데이터 베이스 설계


ROOM_PARTICIPANTS
room_id int NN
user_id int NN
role varchar(20)
joined_datetime datetime
GAME_ROOMS
room_id int
title varchar(100)
host_user_id int NN
status varchar(20)
created_datetime datetime
max_participants int NN
is_private char(1) NN
GAME_HISTORY
game_id int
room_id int
black_user_id int
white_user_id int
winner_id int
start_datetime datetime NN
end_datetime datetime
GAME_MOVES
move_id int
game_id int NN
move_order int NN
x_coord int NN
y_coord int NN
color varchar(10) NN
played_datetime datetime
USER_REPORTS
report_id int
reporter_id int NN
target_id int NN
reason varchar(500)
status varchar(20)
created_datetime datetime
USERS
user_id int
email varchar(100)
password varchar(255) NN
nickname varchar(50) NN
face_encoding text
profile_image varchar(255)
is_social_login char(1) NN
social_type varchar(20)
social_id varchar(100)
role char(1) NN
is_active char(1) NN
joined_date datetime NN
last_login_datetime datetime

1 USERS

- USER_ID (PK): 사용자 고유 번호(NUMBER, 자동 생성)
- EMAIL, PASSWORD, NICKNAME: 로그인, 기본정보
- IS_SOCIAL_LOGIN, SOCIAL_TYPE, SOCIAL_ID: 소셜 정보
- PROFILE_IMAGE: 프로필 이미지 주소
- ROLE: 사용자 역할 (C: 일반, A: 관리자, G: 게스트, B: 봇)
- IS_ACTIVE: 계정 활성 여부
- JOINED_DATE, LAST_LOGIN_DATETIME: 가입, 접속 이력
- FACE_ENCODING: 얼굴 이미지 인코딩

2 GAME_ROOMS

- ROOM_ID (PK): 게임방 고유 번호 (NUMBER, 자동 생성)
- TITLE, HOST_USER_ID (FK): 방 제목, 개설자
- STATUS: 상태 (WAITING, PLAYING)
- MAX_PARTICIPANTS: 최대 인원
- IS_PRIVATE: 비공개 여부

3 ROOM_PARTICIPANTS

- ROOM_ID, USER_ID (PK): 복합키
- ROLE: 참가 유형 (CHALLENGER, SPECTATOR)
- JOINED_DATETIME: 입장 시간

4 GAME_HISTORY

- GAME_ID (PK): 게임 고유 번호 (NUMBER, 자동 생성)
- ROOM_ID (nullable FK): 게임이 열린 방
- BLACK_USER_ID, WHITE_USER_ID : 참여자(흑, 백)
- WINNER_ID: 승자
- START_DATETIME, END_DATETIME: 게임 시작, 종료 시각

5 GAME_MOVES

- MOVE_ID (PK): 수 고유 번호 (NUMBER, 자동 생성)
- GAME_ID (FK): 어느 게임에서 발생한 수인지
- MOVE_ORDER: 몇 번째 수인지
- X_COORD, Y_COORD: 바둑판 상 위치
- COLOR: 흑/백
- PLAYED_DATETIME: 수를 둔 시간

6 USER_REPORTS

- REPORT_ID (PK): 신고 고유 번호 (NUMBER, 자동 생성)
- REPORTER_ID, TARGET_ID: 신고자와 대상자
- REASON: 신고 텍스트
- STATUS: 신고 처리 상태 (PENDING, RESOLVED 등)
- CREATED_DATETIME: 신고 시각

핵심 기술 스택

구분	기술
백엔드 (Backend)	Java 21, Spring Framework, Spring MVC, BCryptPasswordEncoder, Tomcat 10.1, MyBatis, HikariCP
AI 서버 (Backend)	Python, Flask, TensorFlow, DeepFace
프론트엔드 (Frontend)	JSP, JSTL, JavaScript (ES6+), WebSocket, Fetch API
데이터베이스	Oracle Database
API 연동	Kakao API (OAuth 2.0 & Link), REST API (RestTemplate)

실시간 대전 시스템

- **목표**

  플러그인 없이 웹 브라우저만으로
  지연 없는 오목 대전 및 채팅 구현

- **기술**

  JSR-356(Java WebSocket API)
  @ServerEndpoint, @OnOpen,
  @OnMessage, ConcurrentHashMap

- **작동 방식**

  JavaScript(Web): 게임방 접속 시 WebSocket 자동 연결

```
new WebSocket(`wss://.../ws/game/${roomId}`);
```

  Java(Server): 각 클라이언트와 1:1 Session 생성 후 처리

  JSON 기반 메시지 처리 (GAME_START, CHAT 등)

```
switch(msg.type) {
    case 'ROOM_STATE': updateLobby(msg); break;
    case 'CHAT': appendChatMessage(msg.senderId, msg.sender, msg.content); break;
    case 'GAME_START': handleGameStart(msg); break;
    case 'UPDATE_STATE': handleUpdateState(msg); break;
    case 'GAME_OVER': handleGameOver(msg); break;
}
```

강화학습 AI 연동

- **목표**

  AI와 실시간 대국 가능

- **기술 구조**

  Spring Server → 게임 로직, 요청 전달

  Python Server (Flask) → AI 추론 연산 전담

  통신 : Spring RestTemplate ↔ Flask REST API

- **AI 학습**
  - 입력: (15, 15, 2) 배열 (자신/상대 돌 위치 분리)
  - 자가 대국(Self-Play) 방식으로 Q-Value 학습
  - Replay Buffer, Best Model 저장 → best.h5 사용
- **예측 흐름**
  - Java → Python /predict 요청 (현재 판 상태 전송)
  - Python → AI 수 예측 → 최적 좌표 응답 (JSON)
  - Java → WebSocket으로 게임판 갱신 전송

얼굴 인식 로그인

- **목표**

  비밀번호 없이, 얼굴로 안전하고 편리하게 로그인

- **기술 스택**

  DeepFace, VGG-Face 모델 내장
  OpenCV

- **얼굴 등록 과정**
  - 입력: (15, 15, 2) 배열 (자신/상대 돌 위치 분리)
  - 자가 대국(Self-Play) 방식으로 Q-Value 학습
  - Replay Buffer, Best Model 저장 → best.h5 사용
- **예측 흐름**
  - 웹캠 촬영 + 이메일 입력
  - DB에서 기존 인코딩 불러옴
  - Python 서버 /verify-face API 호출
  - 두 인코딩의 유클리드 거리 계산
    - 거리 < 0.6 → 동일인 인정 → 로그인 성공

프로젝트 일정

프로젝트 기간: 2025년 6월 2일 ~ 2025년 6월 30일 (총 4주)

구분	1주차	2주차	3주차	4주차
1. 기획/설계	요구사항, 아키텍처 설계			
	DB 모델링			
2. 백엔드 개발		Spring 설정, 기본 CRUD, 인증 구현		
		WebSocket 기반 실시간 로직 구현		
3. 프론트 개발		기본 UI/UX 개발		
		추가 UI 개발 (관리자, 얼굴 인식 페이지)		
4. AI 연동		Python AI 서버 구축 및 API 구현		
			Spring-Python 서버 간 연동	
5. 통합/테스트			테스트 및 버그 수정	
				발표 자료 준비 및 시연 영상

주요 기술적 과제 및 해결 방안

- **과제**
  실시간 상태 동기화 및 데이터 정합성 문제
- **원인**
  1. 사용자가 비정상적으로 접속을 종료할 경우
  2. 서버의 실시간 메모리 상태와 데이터베이스의 영구 데이터 간의 불일치가 발생
  3. NullPointerException이나 참조 무결성 오류로 이어짐
- **해결**
  - 모든 상태 변경의 최종 결정권은 실시간 상태를 정확히 아는 GameWebSocketHandler가 갖도록 설계
  - DB 작업은 서비스 계층에 위임하되 그 실행 시점은 웹소켓 핸들러가 명확히 제어

주요 기술적 과제 및 해결 방안

- **과제**

  Java-Python 서버 간의 통신 및 데이터 변환

- **원인**

  1. Spring(Java) 서버와 AI(Python) 서버는 서로 다른 언어와 환경에서 동작.
  2. Java의 2차원 배열(int[][])을 Python의 numpy 배열로, 다시 Python의 예측 결과를 Java의 Point 객체로 변환
  3. 데이터 형식 불일치나 직렬화/역직렬화 오류가 발생할 수 있음

- **해결**

  - 두 서버 간의 통신 프로토콜로 HTTP 기반의 REST API를 채택하고, 데이터 교환 형식은 언어에 독립적인 JSON으로 표준화
  - Spring의 RestTemplate과 Python의 Flask를 활용하여 안정적인 통신 시스템을 구축

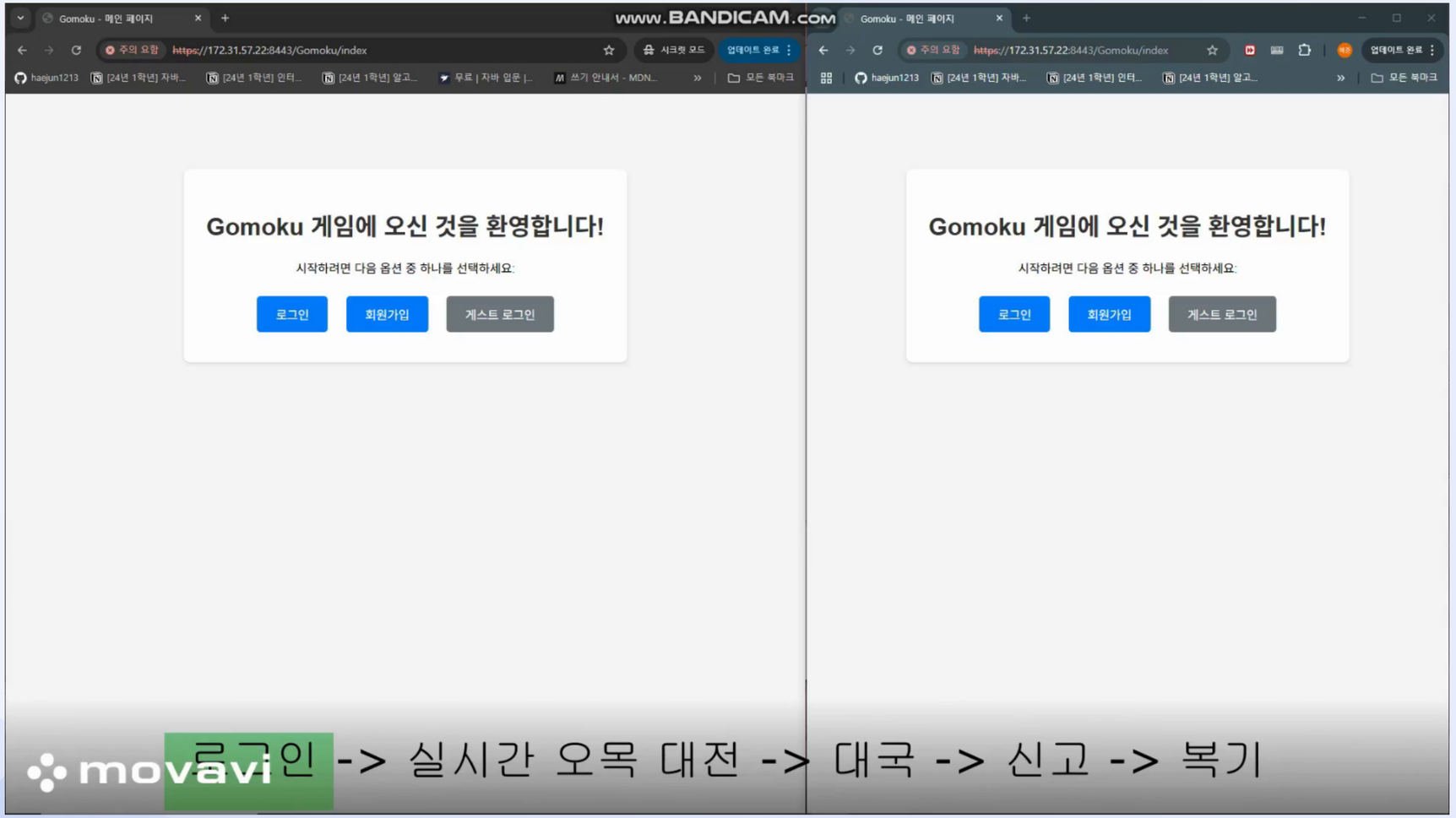
시나리오 요약

01. 로그인 -> 대전 방 입장 -> 카카오 초대 -> 대국 진행 -> 신고 -> 마이페이지 -> 복기

02. 관리자 로그인 -> 유저 목록 조회 -> 방 목록 조회 -> 신고 처리

03. 얼굴 로그인, AI 대전, 프로필 이미지

시연 영상


www.BANDICAM.com
Gomoku - 메인 페이지
주의 요함
https://172.31.57.22:8443/Gomoku/index
시크릿 모드
업데이트 완료
Gomoku 게임에 오신 것을 환영합니다!
시작하려면 다음 옵션 중 하나를 선택하세요:
로그인
회원가입
게스트 로그인
movavi
로그인 -> 실시간 오목 대전 -> 대국 -> 신고 -> 복기

향후 개선 사항

01.

오목 AI 고도화

현재 DQN 기반의 AI 모델을, MCTS(몬테카를로 트리 탐색)를 결합한 AlphaZero 아키텍처로 업그레이드하여 인간의 한계를 뛰어넘는 기력을 구현

02.

랭킹 및 티어 시스템 도입

사용자의 승패에 따라 점수를 부여하고, 브론즈, 실버, 골드와 같은 티어 시스템을 도입하여 사용자들의 경쟁심과 성취감을 자극

03.

여러 규칙 도입

현재 경기 규칙은 자유룰 하나지만 렌주룰이나, 오프닝룰 등의 다른 경기 규칙도 추가

프로젝트 후기

01.

실전에서 부딪힌 문제들

- WebSocket 동시성 제어
  - 다중 접속 중 세션 충돌 방지 → ConcurrentHashMap
- DB 트랜잭션 및 교착 상태
  - 게임 중복 저장 방지, 실패 시 롤백 처리 등 트랜잭션 실습
- HTTPS 보안 정책
  - Self-signed Certificate 설정으로 webcam 사용 환경 구축
- DTO 설계 경험
  - 다양한 JSON 메시지를 구조화하기 위해 DTO 클래스(예: MoveMessage, ChatMessage)를 정의하고, ObjectMapper로 매핑 처리

02.

학습 포인트

웹소켓, HTTP, DB, AI까지 아우르는 Full-stack 시스템 직접 설계

단순 기능 구현을 넘어, "왜 이렇게 설계해야 하는가"를 고민하며 확장성과 안정성을 고려한 구조 구현

딥러닝 및 강화학습 모델을 서버와 연동하며, AI 추론 → API 설계 → UI 반영까지의 전체 흐름을 완전하게 연결

03.

아쉬움과 앞으로의 목표

DQN 학습 구조나 보상 설계 등 강화학습 이론에 대한 수학적 이해 부족
→ AI 성능 최적화에 아쉬움

다음 프로젝트에서는 AI 모델 자체의 성능까지 더 깊게 튜닝 가능한 역량을 갖추고 싶음

클라우드 기반 배포(AWS, CI/CD) 및 모델 서빙도 학습 예정

감사합니다