YamYam

현명은, 구자현, 박동진, 조현민

목차

YamYam 팀소개

YamYam

이번 윈터데브캠프기간중에
배운것들을 열심히 흡수하겠다.
그만큼 프로젝트를 씹어먹겠다.

구자현	박동진	조현민	현명은
BackEnd개발자	BackEnd개발자	Client개발자	Client개발자

NoPOKER 소개

“인디언 포커 하죠?”

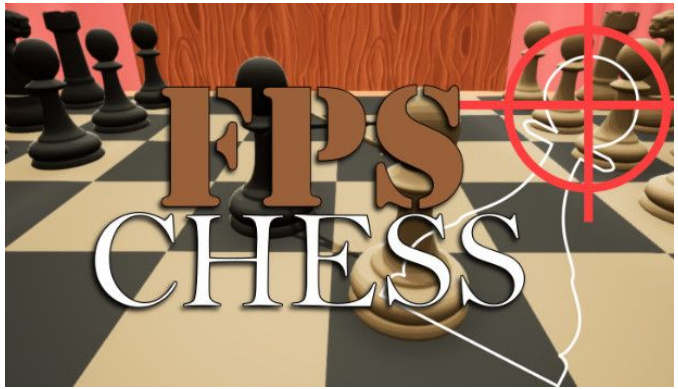
NoPOKER 소개

게임계의 평양냉면

NoPOKER 소개

몬가.....
MSG가 필요해

NoPOKER 소개



체스인척 하는
FPS게임

이걸 인디언 포커에?
당장하자!

NoPOKER 소개

게임 컨셉

인디언 포커를 기반으로 한 난투형 액션 게임

게임 규칙

- 인디언포커를 진행한다
- 인디언포커가 끝난 후, 플레이어간 전투를 한다.
- 단, 인디언 포커에서 숫자가 높은 사람이 좋은 무기를 가질 수 있다.
- 이 라운드에서 모든 상대방을 쓰러뜨린 경우 베팅된 금액을 모두 가져간다.
- 파산하지 않은 최후의 1인만 남는 경우 게임이 종료된다.

팀 목표

현재 팀의 상태

- 개발이슈 관련 얘기가 서로 부담스러운 상태
- 회의 할때 오디오 공백이 많이 생기는 상태
- 질문하는게 눈치보이는 상태
- 자신이 구현해야할 작업에 대해 자신이 없어 대답을 잘 못하는 상태
- 서버 기반 게임 구현을 해본 적 없음
- 복잡한 게임을 구현하기 위해 팀의 포텐셜을 끌어올려야 함

팀 목표

이슈를 조기발견하여 이슈 해결 비용을 줄일 수 있는 팀

- 화목하고 자유로운 분위기
- 말하는 것에 있어서 부담갖지 않는 팀
- 이슈를 자유롭게 공유할 수 있는 분위기
- 서로의 생각을 존중하고 수용할 수 있는 분위기를 가진 팀
- 자신이 작업한 것에 대해 확신있게 말할 수 있는 팀
- 제한 시간내에 목표 요구사항을 구현할 수 있는 팀

점검

- 정기회의때, 주간에 있었던 이슈에 대해 허심탄회하게 이야기하는 시간을 가짐
- 이야기 하는 시간에 그랬구나 대화법을 사용해 서로 감정이 상해있는 부분은 없는지 체크

YamYam’s workflow

데일리 스크럼

- 9:30 ~ 10:00
- 건강, 한 일, 할 일, 이슈, 질문

OO월 OO일 스크럼

이름	한 일	할 일	이슈	건강상태	질문

건강 상태 파악을 통해 팀원 태스크 진행속도 가늠

작업 상황 공유를 통한 동기 부여

이슈, 질문을 매일 작성하며 자유로운 분위기 형성

YamYam’s workflow

GROUND RULE

- 코어타임 : 평일 13:00 ~ 15:00
- 화날때 ‘용용체’쓰기
- 의사결정 방식 : 사다리 타기
- 읽씹 안하기, 읽었다는 표현해주기
- 잡담나누기

자유로운 팀 분위기 유지

YamYam's workflow

정기회의

주간 회고 + 주간 회의
금요일 20:00, 강의 있으면 강의 끝나고

회의내용

- 개발 상황 공유
- 개발일지 작성

준비할 것

- 개인 회고록
- 회의해볼 필요가 있는 사항
- 일주일간 개발 상황
- TodoList 작성

회의 Deadline까지 해야한다는 동기 부여

To Do List 작성을 통해 남은 태스크 파악

비대면으로 인한 작업상황 온라인 공유 필요

목소리로 대화하는 것의 중요성

개인 목표 - 현재상황

현명은	- 비동기(스레드)에 대한 이해가 부족한 상태 - 다른사람이 나의 코드를 보는것이 두려움
조현민	- 서버 및 네트워크에 대한 지식이 얕음 - 코드 작성시에 어떻게든 작동만 하면 된다는 마인드였지만 이제 바꾸고 싶음
박동진	- 라이브러리(개발도구)를 생각 없이 사용(구글링으로 사용법만 공부) - 테스트 코드 작성 경험 거의 없음
구자현	- 성능 시나리오 작성 경험이 없음 - 코드를 작성하다 나중에 보면 의도를 기억하지 못함

개인 목표

현명은

- 유니티에서 비동기프로그래밍 방식으로 코드를 구현할 수 있는 개발자
- 코드리뷰를 두려워하지 않는 개발자

조현민

- NoPOKER의 게임 네트워크에 대해 설명할 수 있는 사람
- 재사용 가능한 코드를 작성하려는 습관을 가진 개발자

박동진

- 개발도구의 공식 레퍼런스를 보고 사용법을 스스로 익힐 수 있는 개발자
- 습관적으로 단위 테스트 코드를 작성하는 개발자

구자현

- 성능을 측정하고 개선이 필요한 부분을 짚어내 개선할 줄 아는 개발자
- 코드에 나의 의도가 명확하게 드러나게 작성하는 개발자

기능 정의

인증 서버 구현
회원가입/로그인화면 구현

채팅 서버 구현
채팅 UI 구현

로비 서버 구현
로비 구현

게임 서버 구현
2D 포커게임 구현

실시간 게임 서버 구현
3D 액션게임 구현

기능 정의 - 인증

일반 사용자

- 회원가입
- 이메일 인증
- 로그인
- 아이디 찾기
- 비밀번호 찾기
- 회원 탈퇴
- 마이 페이지
- 친구 추가, 수락
- 친구의 상태 확인
- 특정 사용자 신고
- 특정 사용자 차단
- 팀 초대

관리자

- 로그인
- 사용자 조회
- 사용자 삭제
- 사용자 수정(이용 정지(Ban))
- 채팅 로그 확인
- 신고 내역 확인

기능 정의 - 채팅

로비채팅

사용자가 접속한 이후의
채팅 내역만 조회 가능

1대 1 채팅

모든 채팅내역 조회가능

팀 채팅

- 게임방에서 팀 간의 채팅 가능
- 매칭 되기 전까지 채팅 내역 조회 가능

기능 정의 - 매칭

솔로 매칭

- 패널티 유저 확인
- 2인/ 4인 매칭
- 먼저 매칭을 돌린 사람이 매칭 우선순위
- 매칭 거절 3회 이상 시 패널티 부여(5분 ~)

팀 매칭

- 친구초대
- (4 - 인원수) 게임 매칭

기능 정의 - 2D 포커

사용자

- 배팅(다이/콜/지정 개수)
- 게임방 나가기
- 게임 결과 확인
- 감정표현

비실시간 네트워크

- 배팅금액 동기화
- 플레이어 순서 동기화

UI

- 상대 플레이어 턴 제한시간
- 게임 판, 턴 횟수
- 전체 플레이어 정보 (닉네임 ,보유 칩)
- 전체 배팅금액
- 게임버튼 UI (배팅, 나가기)
- 감정표현

기능 정의 - 3D 액션

게임 로직

- 플레이어 움직임
- 공격 및 체력 시스템
- 히트박스 설정
- 카드 별 무기 밸런스
- 베팅 금액

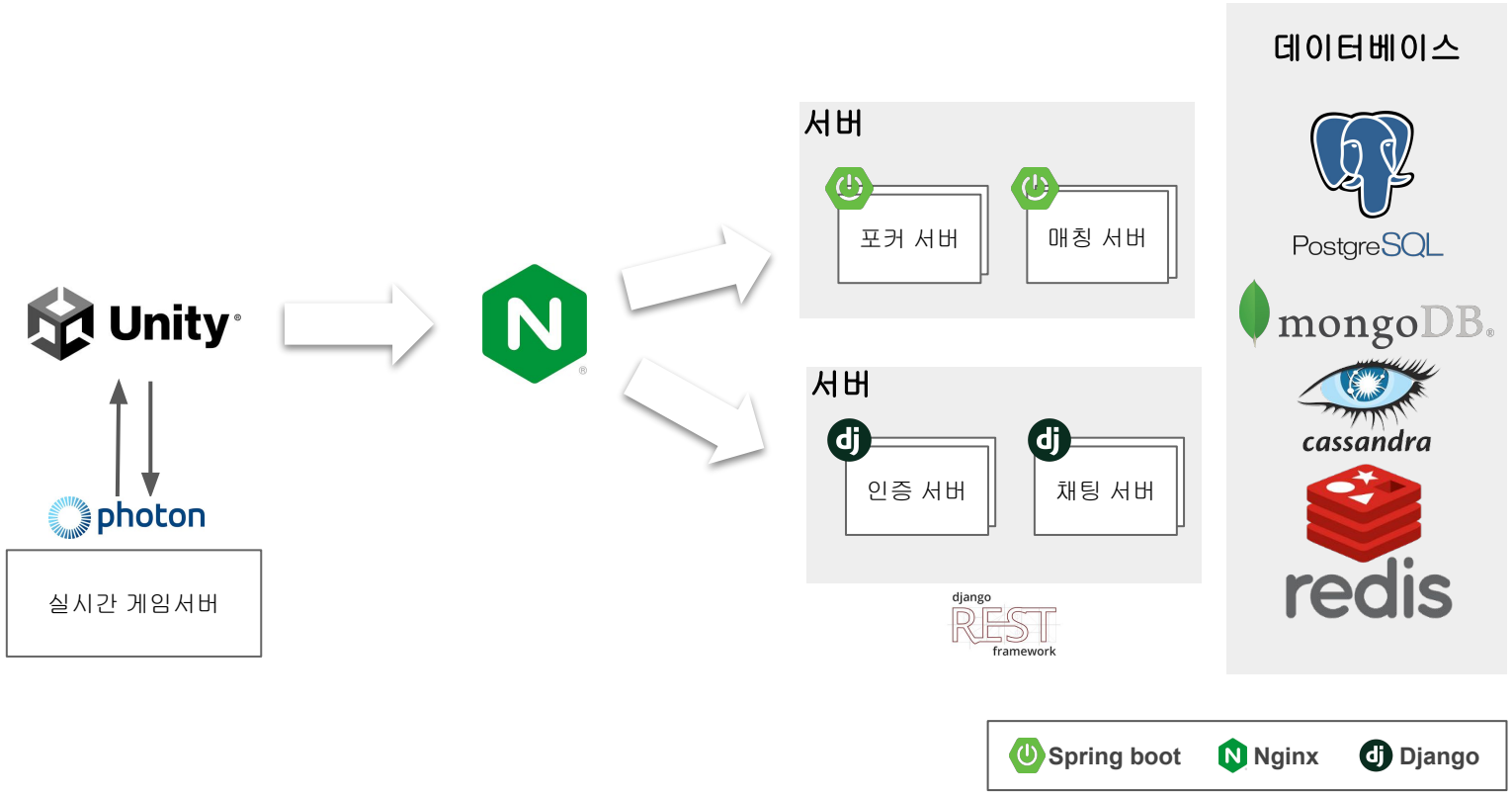
실시간 네트워크

- 움직임 동기화
- 체력 및 피격 동기화

UI

- HUD
- 레벨디자인

NoPOKER 아키텍처



아키텍처

API Gateway

- 인증 API를 호출하면 이후의 API호출에 필요한 토큰을 돌려 받음
- 이후에 사용자가 이 토큰을 가지고 원하는 API를 호출하면 API Gateway는 Access KEY는 검사하지 않고 이 토큰이 맞는지만 체크한 후 해당하는 서비스를 호출
- 원하는 서비스를 호출하는 방식은 사용자가 호출할 서비스와 내부에서 개발된 엔드포인트를 테이블 형태로 가지고 있다가 사용자가 호출한 서비스를 비교해서 내부의 엔드포인트로 바꿔서 호출하고 그 리턴 값을 사용자에게 돌려주는 방식사용

아키텍처

API Gateway

테이블로 관리

외부로 알려지는 서비스 URL

내부 URL

필요에 따라 내부 URL을 바꿔서 응답 가능

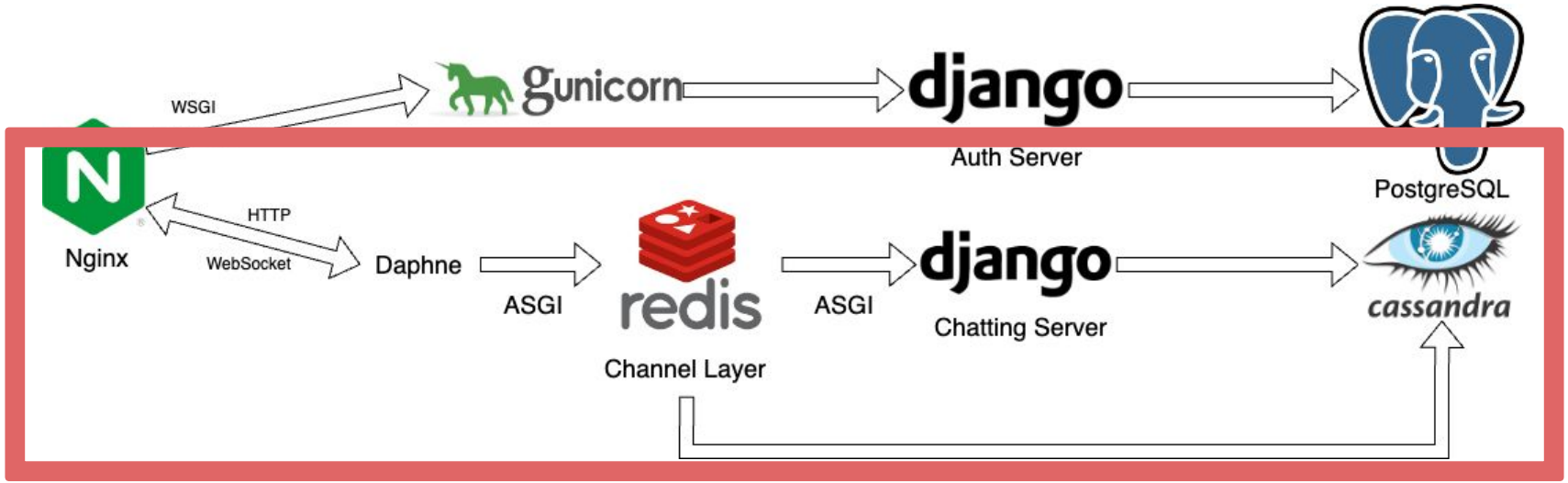
아키텍처

API Gateway

다양한 내부의 서비스에 대한 관리 편리성과
공통으로 개발해야할 부분을 API Gateway가 해결 가능

Nginx를 활용해 API Gateway 구현

NoPOKER 채팅 아키텍처



- HTTP와 WebSocket handling을 위해 django channels 사용
- Daphne : ASGI Server, HTTP/HTTP2 및 WebSocket 프로토콜 서버(gunicorn과 유사)
- Channel Layer : 여러 사용자가 채팅을 할 수 있도록 해줌, Redis를 백업 저장소로 사용
- chat log를 Cassandra에 저장 (Cassandra는 RDB보다 더 나은 write perfomance를 보여줌)

Milestone

1. 서비스 구현 및 서버 연결 (~ 2월 중순)

인증	채팅	로비	게임	실시간 게임
인증 서버 구현 회원가입/로그인화면 구현	채팅 서버 구현 로비, 대기방 화면 구현	로비 서버 구현 로비 구현	게임 서버 구현 2D 포커게임 구현	실시간 게임 서버 구현 3D 액션게임 구현

2. 성능 개선 및 기능 추가

- 성능 프로파일링 및 개선 작업
- 신고, 차단기능

업무분담 및 개발계획

현명은

- 로비
- 2D 포커게임
- 로그인/회원가입

~ 1.27 : **화면설계 및 구현**

1. 회원가입/ 로그인 화면 구현
2. 매칭화면 구현
3. 2D포커게임 화면구현

~ 1.30 : **클라이언트 백엔드 연결**

1. 회원가입/로그인 백엔드 연결
2. 매칭 백엔드 연결

~ 2.7 : **게임기능 구현 및 연결작업**

1. 2D게임 구현
2. 2D게임 백엔드 연결
3. 2D게임 3D게임 전환

~ 2월중순 : **성능개선, 프로파일링 및 개선작업**

조현민

- 실시간 서버
- 3D 액션게임

~ 1.18 : **레벨 디자인**

1. Lo-Fi Prototype
2. 유니티 적용

~ 1.30 : **화면설계 및 구현**

1. 3D게임 화면구현
2. 3D게임 기능구현

~ 2.13 : **게임기능 구현 및 실시간 서버 연결**

1. 실시간 서버 연결
2. 2D게임 백엔드 연결

~ 2월중순 : **성능개선, 프로파일링 및 개선작업**

업무분담 및 개발계획

박동진

- 인증 서버
- 채팅 서버

~ 1.27 : **인증 서버 구현**
- -인증 API
- -친구 API
- -신고 API
- -초대 API
- -관리자 API

~ 2.10 : **채팅 서버 구현**
- -로비 채팅
- -1 대 1 채팅
- -팀 채팅

~ 2월중순 : **성능개선, 프로파일링 및 개선작업**

구자현

- 로비 서버
- 2D게임 서버

~ 1.27 : **로비 api 구현**

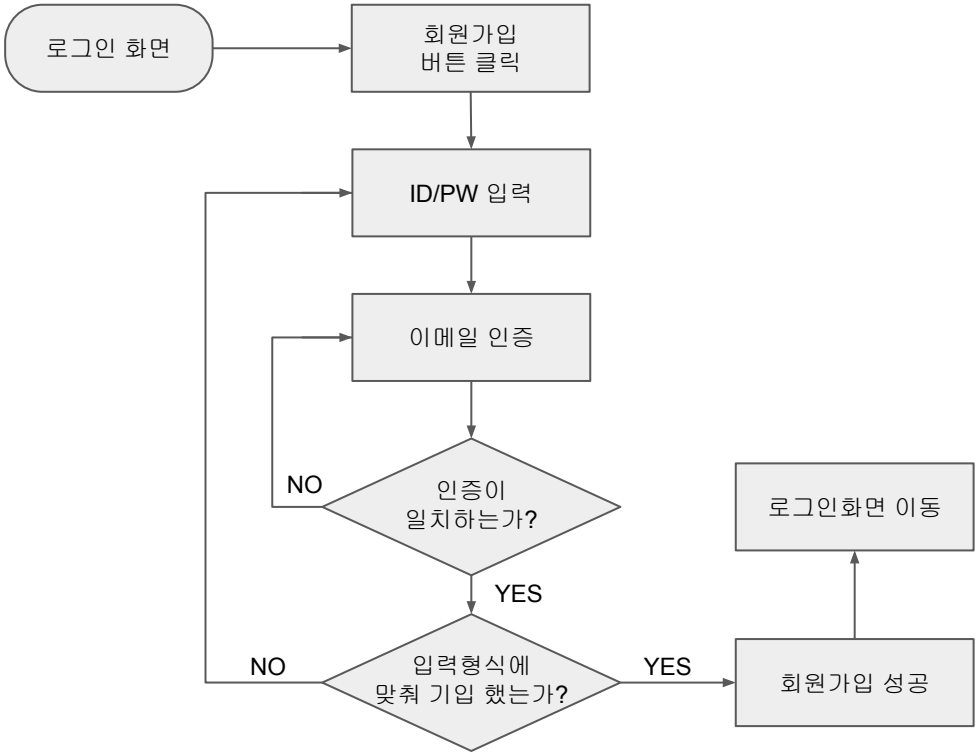
~ 2.7 : **2D 게임 서버 구현**

~ 2월중순 : **성능개선, 프로파일링 및 개선작업**

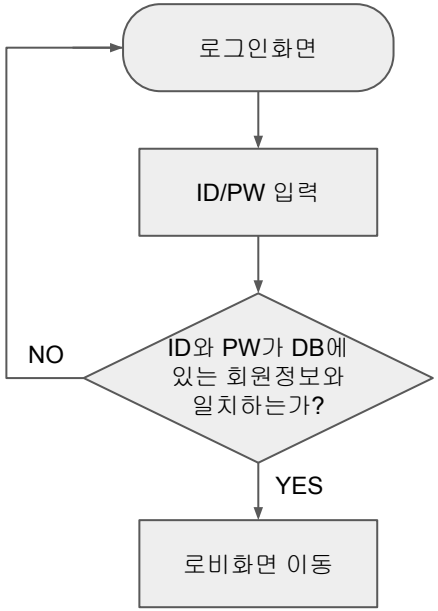
Appendix

Appendix - ERD

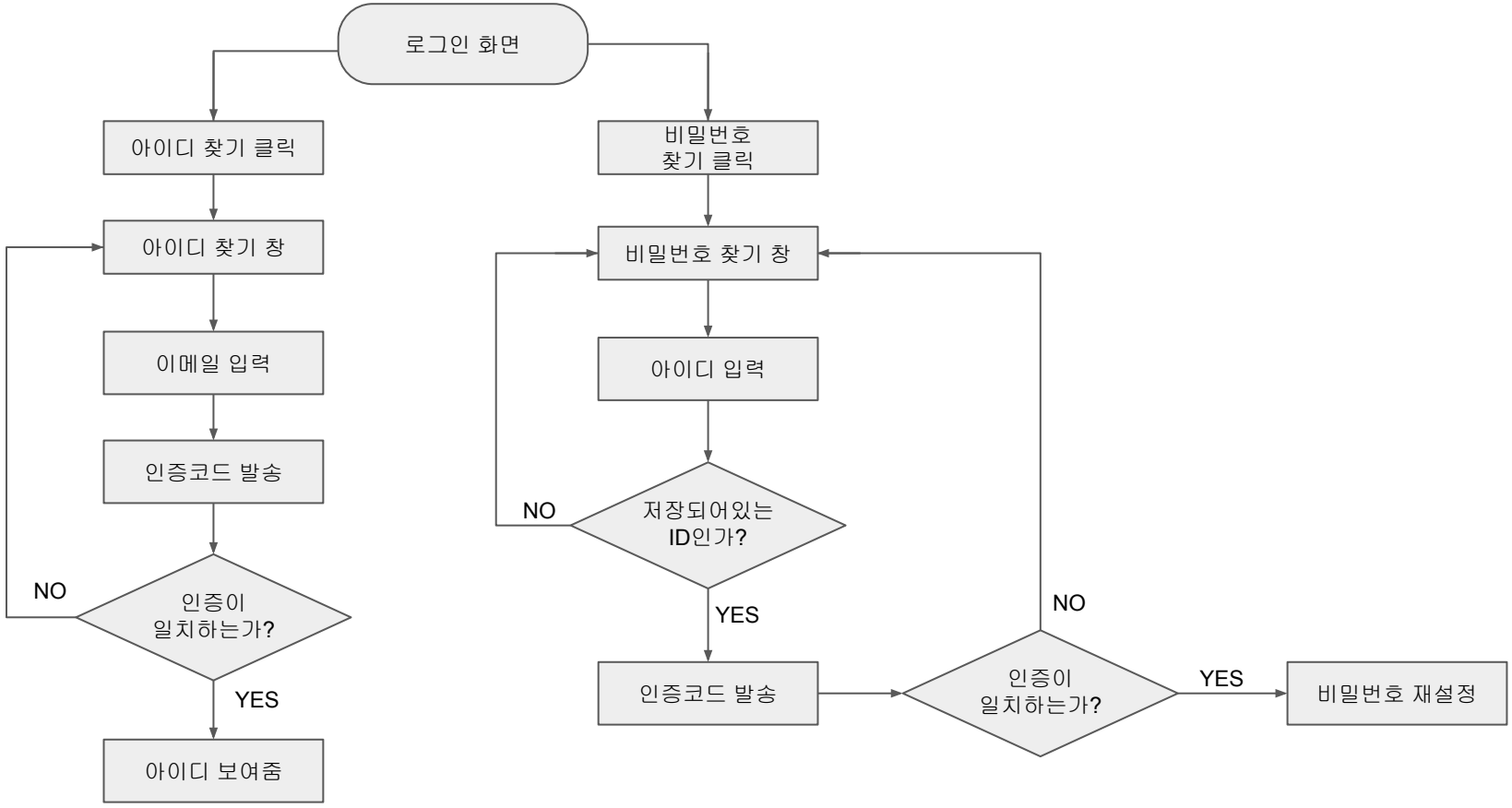
Appendix - 흐름도 (회원가입 & 로그인)



회원가입

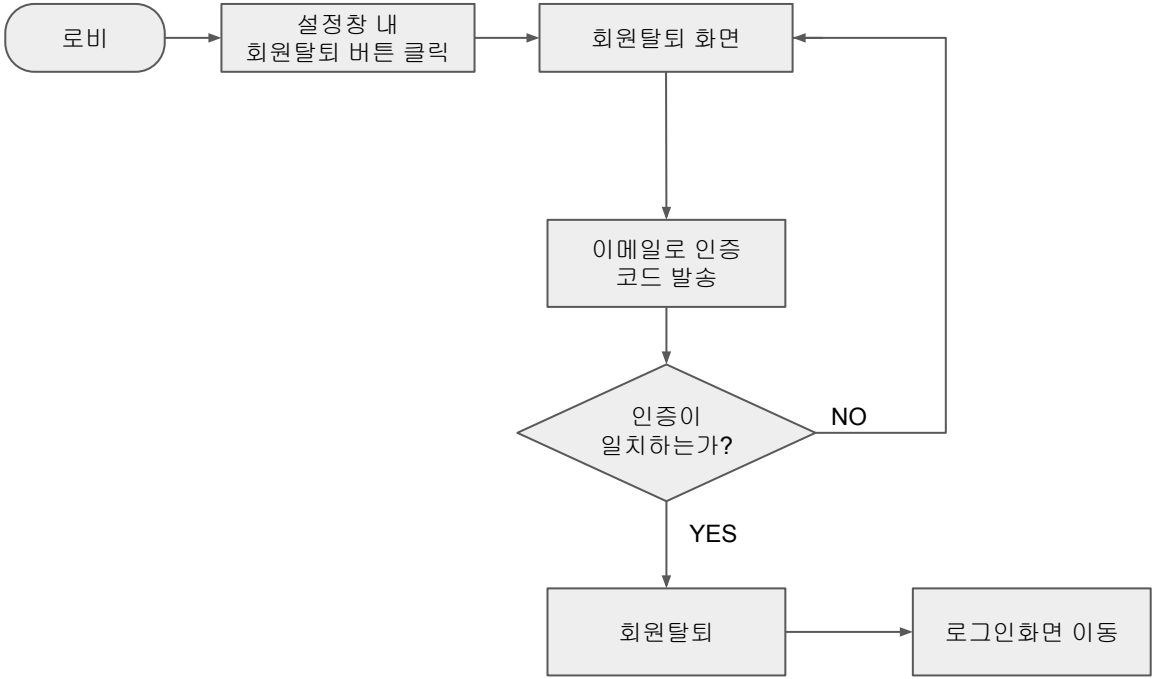


로그인

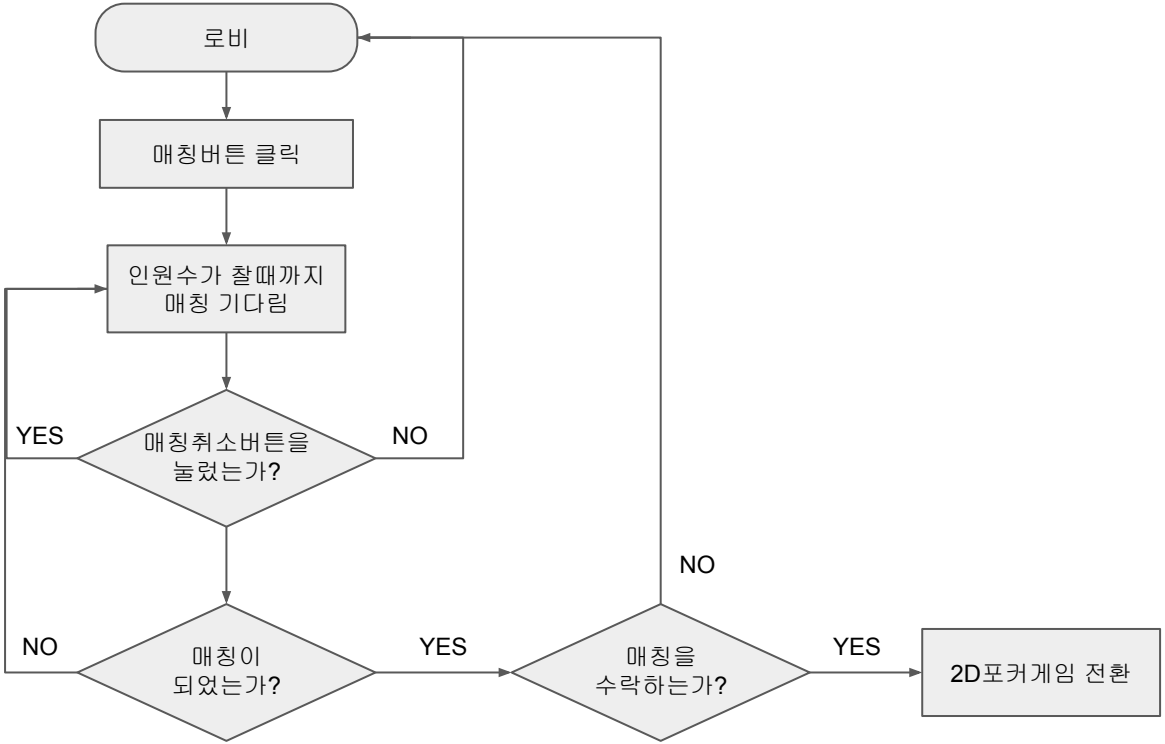

로그인 화면
아이디 찾기 클릭
아이디 찾기 창
이메일 입력
인증코드 발송
인증이
일치하는가?
NO
YES
아이디 보여줌
비밀번호
찾기 클릭
비밀번호 찾기 창
아이디 입력
저장되어있는
ID인가?
NO
YES
인증코드 발송
인증이
일치하는가?
NO
YES
비밀번호 재설정

아이디/비밀번호 찾기

Appendix - 흐름도 (회원 탈퇴)

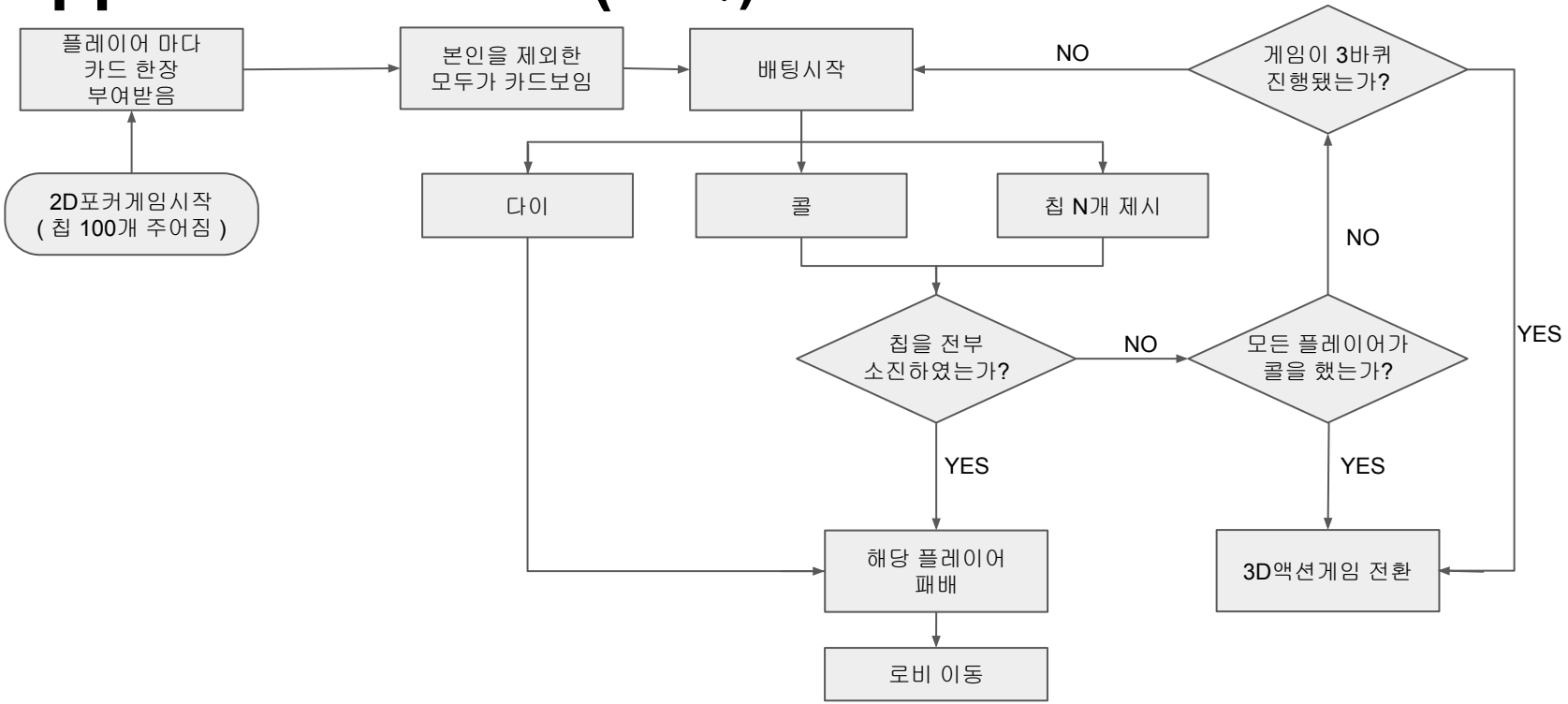


Appendix - 흐름도(매칭)

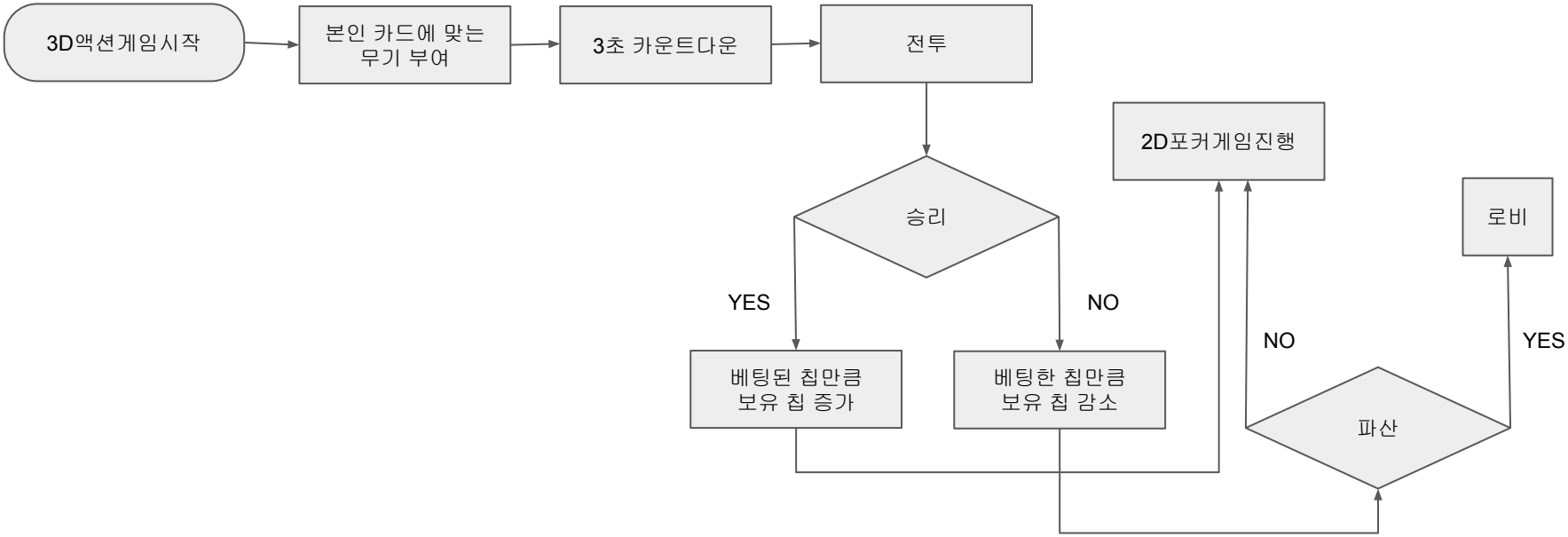


매칭

Appendix - 흐름도(포커)

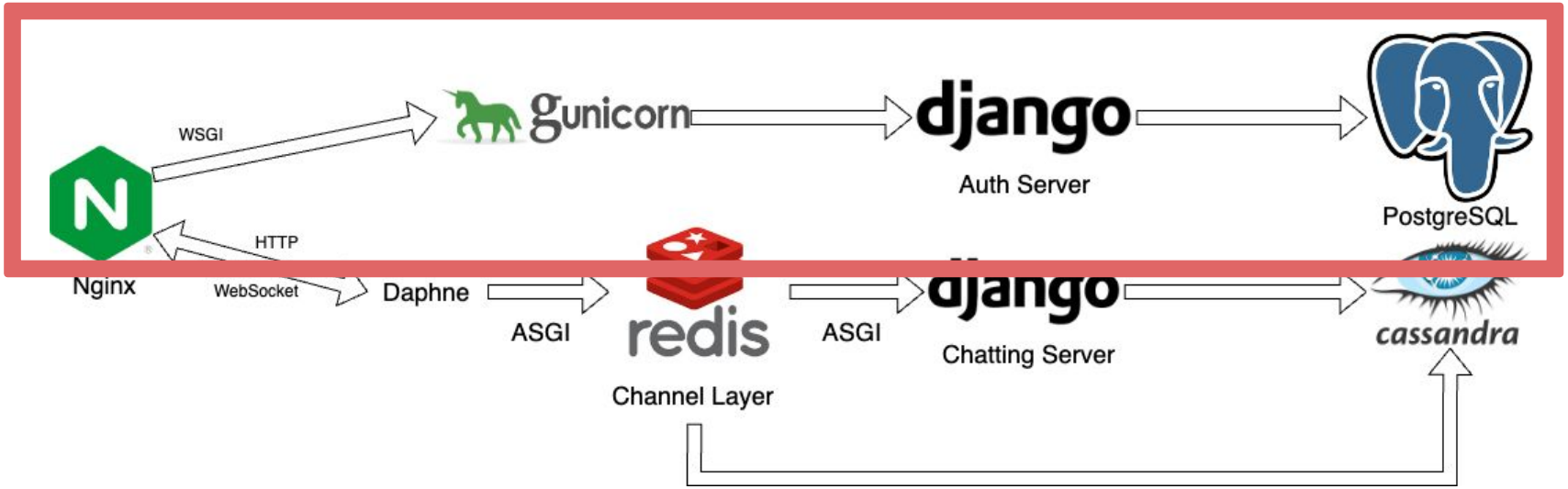


Appendix - 흐름도(액션)

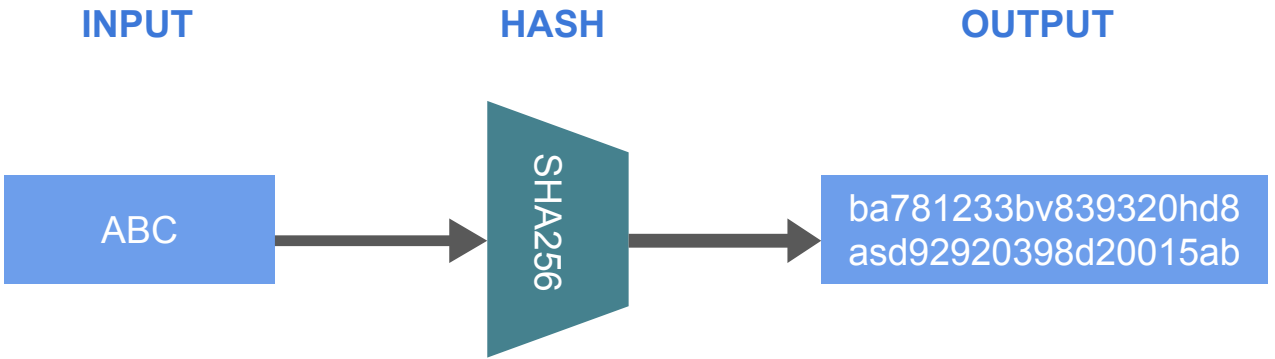


3D 액션 게임

NoPOKER 인증 아키텍처



인증 (회원가입)



- 회원 가입 시 이메일로 인증 링크 발송(SMTP 프로토콜)
- password를 SHA256 방식으로 암호화하여 사용자정보를 RDB에 저장(PostgreSQL)

인증 (로그인)

로그인 시, Auth Server에서 Access Token, Refresh Token 발급 (JWT 기반)
이때, Refresh Token은 Redis에 저장
Access Token의 유효 기간은 6시간, Refresh Token의 유효 기간은 2주

인증 (로그인)

클라이언트는 httpOnly cookie에 Token저장

CSS(Cross Site Script) 스크립트 기반 공격을 방어

인증 (token handler)

- 인증을 요하는 API 요청을 할 때, API Gateway에서 인증 서버의 token handler로 proxy pass

Access Token이 유효	Return HTTP Status 200
Access Token이 유효x **Refresh Token이 유효** **Refresh Token이 DB에 존재**	Access Token, Refresh Token을 재발급 Return HTTP Status 200 (새로운 Refresh Token은 Redis에 저장, 기존 Refresh Token은 Redis에서 삭제 = **Refresh Token Rotation**)

Refresh Token Rotation

- Refresh Token은 Access Token을 한번만 재발급해줄 수 있음 (일회용)
- Refresh Token 탈취 시, 리스크를 줄일 수 있음

인증 (token handler)

- 인증을 요하는 API 요청을 할 때, API Gateway에서 인증 서버의 token handler로 proxy pass

Access Token이 유효x **Refresh Token이 유효** **Refresh Token이 DB에 존재x**	Return HTTP 401 Unauthorized ERROR
Access Token이 유효x **Refresh Token이 유효x**	Return HTTP 401 Unauthorized ERROR

인증 (token handler)

- token handler의 return 결과에 따라 API Gateway는 요청한 API에 proxy pass를 해줄지 결정

HTTP Status 200	요청한 API로 proxy pass
HTTP Status 401	Client에 401 error Return

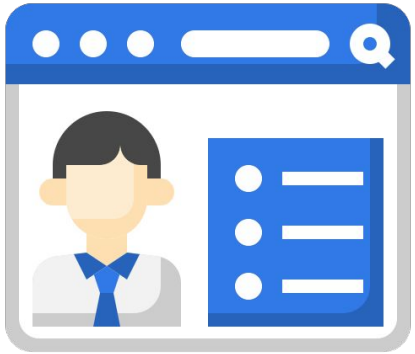
인증 (아이디 찾기, 비밀번호 찾기)

- 이메일 발신(SMTP 프로토콜)
- 아이디 찾기 : 이메일과 연동된 아이디 RETURN
- 비밀번호 찾기 : 비밀번호를 재설정 할 수 있음

인증 (회원 탈퇴)

- 이메일로 인증 링크 발송(SMTP 프로토콜)
- 사용자가 갖고 있는 Refresh Token을 Redis에서 삭제
- DB에서 사용자 정보 삭제
- Access Token 유효기간은 짧기 때문에 리스크 최소화 가능

인증 (마이페이지)



- 로비의 유저 목록, 채팅에서 특정 사용자의 마이페이지 접근 가능
- 이름, 가입 일시, 닉네임, 이메일, 승/패 확인 가능

인증 (친구)

- 친구 테이블 추가
- 친구 추가를 수락해야 친구가 됨
- 친구의 상태 확인 가능
  (온라인, 게임중, 자리비움, 오프라인)

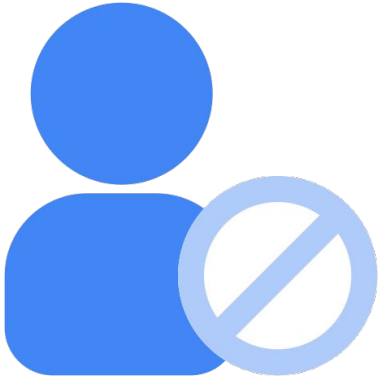
인증 (신고)

- 특정 사용자 신고 가능
- 신고 내역을 관리자가 확인 가능
- 관리자가 신고 내역을 확인하고 사용자를 정지시킬 수 있음

인증 (초대)

- 특정 사용자를 팀에 초대할 수 있음
- 사용자 정보를 DB에서 가져와 매칭 서버로 전달

인증 (차단)



- 특정 사용자를 차단할 수 있음
- 차단한 사용자는 로비에서 보이지 않음
- 차단한 사용자의 채팅은 보이지 않음

인증 (관리자)

- 관리자는 DB에 직접 등록 (회원가입 없음)
- 관리자는 웹 페이지에서 로그인 가능
- 관리자는 사용자 관리 API를 사용할 수 있음

  사용자 조회 | 사용자 삭제 | 사용자 수정

- 관리자는 채팅 로그를 확인하는 API를 사용할 수 있음
- 관리자는 신고 내역을 확인하는 API를 사용할 수 있음
- 관리자 로그인 트래픽은
  몰리지 않을 것이므로 인증은 세션 활용

  세션을 DB에 저장 | 관리자가 API를 사용할 때, DB에 세션있는지 확인

End Of Document