

SeaGull

eGuBa(이거봐)

박성준 박정원 이효승 이범수



01

개인 목표

팀 목표

02

서비스 개요

03

WorkFlow

개발 계획

박정원(FE, web) : Why

- 생산성, 확장성을 확보할 수 있는 컴포넌트 설계하는 능력 기르기
- 협업 시 진행사항 공유를 위해 필요한 문서화 작업역량 기르기
- 동적인 화면 구현 기술 익히기

박정원(FE, web) : How

- 2페이지 이상에서 공통적으로 재사용할 수 있는 체계적인 컴포넌트 만들고 사용해보기
- 개발을 시작하는 날부터 매일 하루에 한번 마무리할 때 구현 진행과정, 사용한 기술을 개발블로그를 통해 정리하기
- 동영상 플레이어를 모바일에서도 대응할 수 있도록 반응형으로 구현해보기

이범수(BE, ai) : Why

- 코드의 유지보수성 강화와 차후의 회고를 위해서 코드의 가독성 개선하기
- 더 나은 성능과 개발효율을 위해서 상황에 따라 더 나은 개발환경을 선택할 수 있는 능력 함양하기

이범수(BE, ai) : How

- 1주 후 주석없이 다시봐도 90% 이상 이해 가능한 코드작성
 - 1) 소스코드 순환복잡도 30이하로 낮추기
 - 2) 주요 함수의 구성을 검증파트, 기능파트로 영역 분할하기
- 일주일에 한번 프레임워크, 라이브러리 사용 이유를 전체 팀 인원 중 75% 이상에게 설득 할 수 있게하기
 - 1) 한 목적을 위한 프레임워크, 라이브러리들을 최소 2개 이상 나열해서 장단점 문서화하기

이효승(BE) : Why

- 어떤것이 좋고 나쁜지 판단하는 능력을 기르는것
- 서버 구조에 대한 전반적인 이해
- 협업을 경험하고 노하우 얻기

이효승(BE) : How

- 나의 결정에 대한 근거 기록하기
- 본격적인 팀 프로젝트 구현 단계 이전에 스프링 프레임워크를 이용한 미니 프로젝트 1개 완성하기
- 팀 워크플로우에 따른 단계별 결과문서 내기

박성준(BE) : Why

- 설계 : 큰 규모의 서비스를 관리하는 서버 관리 기술을 익히는 것
- 기능 : 개념적으로만 배운 서버에서 TCP 통신이 이뤄지는 방식을 직접 구현해보는 것
- 협업 : 팀 단위 개발에 대한 협업 능력을 키우기

박성준(BE) : How

- 설계: Spring Boot 기반 MSA 구조로 서버를 분할하여 관리
- 기능: 많이 사용되고 있는 통신 관련 오픈소스를 활용해, 실시간 통신을 다루는 소켓 API 구현해보기
- 협업: 코드 리뷰를 위해 Git Commit 컨벤션 적용해보기

박정원(FE, web)

- 전체적인 뷰 구성
- Socket을 통한 실시간 채팅 시스템, 영상 공유 원격 스트리밍 추천 알고리즘
- 유저 데이터와 연동

이효승(BE)

- 메인 서버 구조 설계, 구현 (동영상 플랫폼)
- 관리자 페이지 구현
- 채팅 서비스를 기반으로 하는 실시간으로 영상을 공유하는 원격 스트리밍 서비스 구현

박성준(BE)

- 실시간 채팅 시스템 설계 및 구현
- 채팅 서비스를 기반으로 하는 실시간으로 영상을 공유하는 원격 스트리밍 서비스 구현

이범수(BE, ai)

- 유저 인증 및 보안
- 유저 DB설계 및 관리 총괄

박정원(FE, web)

- 기존에 단순히 뷰를 구현하는데 초점이 맞추어져 있었다면 목표 달성을 위해 설계 단계에서 컴포넌트 분리에 집중해 하나의 컴포넌트를 여러 페이지에서 재사용해볼 수 있었고 초반의 계획에서 수정되는 경우에도 유지보수에 용이함을 경험함
- 매일 개발 과정을 기록하지는 못했지만, 매주 작성한 개발일지나 코드리뷰 관련 내용을 꾸준히 정리하면서 문서화의 중요성을 한번 더 깨달음
- 개발 기간 관계 상 '모바일에서도 대응할 수 있는' 반응형 구현이라는 목표는 달성하지 못했지만 다양한 해상도로 이용하고 있는 사용자의 입장에서 편리한 UI를 제공하기 위해 화면의 크기가

박성준(BE)

- 있을 경우까지 생각해 구현해볼 수 있었음
- Stomp, Kafka 두 개로 채팅 서비스를 구현해봄으로써 각각의 장단점을 알 수 있었음
- 방 링크에 따라 서로 다른 실시간 연결을 해주는 과정을 진행해봄으로써 브로드 캐스팅의 개념을 알 수 있었음
- MySQL과 레디스의 사용 용도와 그 성능 차이를 알 수 있었음
- 프론트와 직접 소통하며 웹 소켓 테스트를 해보면서 어떤 부분에서 트래픽이 발생할지에 대한 생각을 할 수 있었음

이효승(BE)

-

이범수(BE, ai)

- 프로그래머의 학습은 어떤 것을 배우기 위해 프로젝트를 하는 것이 아닌, 프로젝트를 진행하다가 부족한 점을 메꾸어 나가기 위해 툴을 학습하는 것이 중요하다는 것을 깨달음
- 위의 깨달음으로 내가 특정한 개발환경을 사용하는 이유를 팀원들에게 설명하는 상황에 큰 도움이 되었음
- 반복문의 사용을 최소화하는 노력덕분에 순환복잡도를 최대한으로 줄일 수 있었음
- 하지만 검증,기능 파트를 따로 분리하는 것에 실패하여 만족할만한 가독성을 갖추는 것에는 실패함

박정원(FE, web)

- React Player 사용중 state를 하나로 묶어 관리했을 때 해당 state내의 하나의 state를 변경해도 다른 값들이 초기화되는 문제
-> 필요한 state값을 별개로 관리하고 나머지는 다 고정된 값으로 처리
- 공유방에서 영상이 공유되고 있는 상태에서 제 3자가 들어올 때 영상이 정상 작동하지 않는 문제
-> 재생 초기값을 재생으로 두고 제3자 입장시 1초에 한번씩 받는 싱크를

받으면 재생유지, 안받으면 일시정지 상태로 처리

- 존재하지 않는 방 링크 접속되는 문제
-> backend에서 방 접속시 get요청으로 방 유무를 값을 받고존재하지 않는

방 링크로 접속할 시 PrivateRoute로 이동해 검증과정을 거친 후 404notfound페이지로

이동

박성준(BE)

- 웹 소켓을 테스트하는 과정에서 포스트맨과 구글 이스텐션을 사용하는데에 어려움을 겪음
-> 테스트용 웹을 구축
- 카프카와 스톱프를 활용하는데에 기술적 어려움을 겪음
-> 스톱프만으로 확실한 기능을 구현
- 레디스 자료구조 동작 구조를 이해 못해 Null값 처리를 못함
-> 직접 RedisTemplate을 사용해 자료구조를 사용함으로써

동작원리를 이해

이효승(BE)

-

이범수(BE, ai)

- 토큰 인증에서 중복로그인 방지문제를 어떻게 해결 할 것인가에 대한 고민을 함
-> 토큰을 DB에 저장하는 방식, 토큰 Fingerprint를 이용하는 방식 둘 다 생각을 해봤지만 이런 경우 stateless 특성의 토큰 인증을 사용할 이유가 없다는 결론이 나오으로써 디스코드처럼 중복 로그인을 허용하는 방식으로 결정함
- 이메일 인증시에 일반 DB를 사용하는 것 말고 성능을 개선할 방안이 있는가에 대한 고민을 함

-> Redis와 Redis 내부의 setex 기능을 사용하여 성능과 메모리

코드통합과 버전관리

깃허브 관리

- 팀 내 깃플로우 전략, 커밋 메시지 컨벤션을 정하고 이에 따라 코드관리



지속적인 소통과 빠른 피드백

스크럼 회의

- 매 시작 전 15분간 회의
- 마무리 일일 회고 작성



협업 경험

유지보수를 고려한 협업

작업 내용 문서화

- 워크플로우에 따른 결과 문서도출
- 결과 문서 양식은 작업 목표, 작업 결과, 다음 계획 수립의 간략한 형식으로 정함



생산성 고려한 개발

팀 내 코드 컨벤션에 따라

Lint를 활용해 코드 작성

- BE:hackday-conventions java guide
- FE: Airbnb JavaScript style guide



NEEDS

- 동적이면서 사용자 편리성을 고려한 UI
- 실시간 영상 공유를 위한 서버 구성
- 동영상 강의를 들었던 경험에서 느꼈던 함께 듣는 사람들과의 소통의 필요성

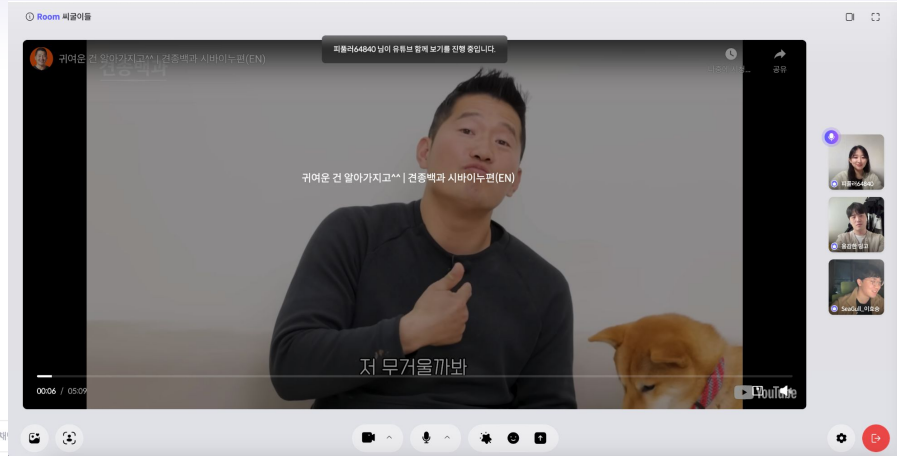
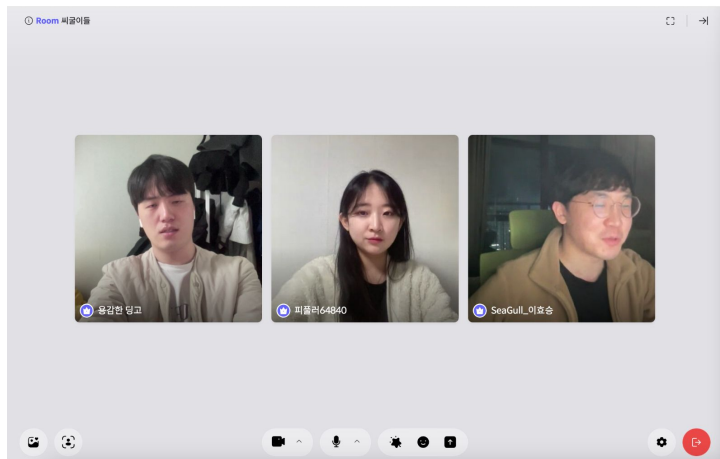


Summary

사람들과의 소통을 실현하는 차별성을 가진 영상 공유 플랫폼

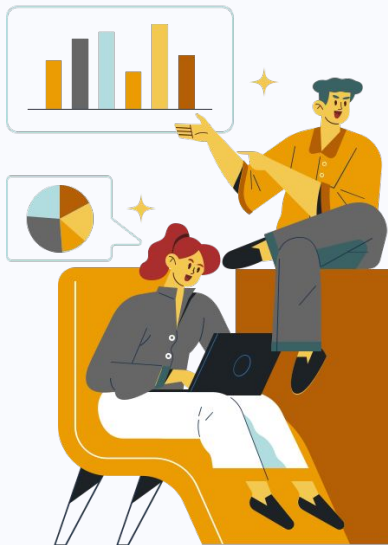
What is eGuBa??

친구들과 함께 하는 영상 공유 플랫폼



+ 차별화된 부가 서비스

원격 동영상 스트리밍
채팅 서비스



Daily Scrum Meeting

- 매일 아침 9시 20분간 회의
- **일일 개발 구현 목표** 설정
- 매일 저녁 9시 회의
- > **노션**을 통한 문서화를 통한 진행 상황 공유



Sprint Review

- 2주 단위 목표 달성을 위한 목표 설정
- **2주 단위로 스프린트** 수행 및 점검
- 개발 중 빠른 피드백을 위한 **개인의 개발 진행과정 문서화**



Git Management

- 팀 내 **커밋메시지 컨벤션**에 따라 커밋 진행
- **깃 플로우 전략**에 따라 branch, develop, master로 구분해서 관리

Ground Rules

그라운드룰

📌 회의시간

대면 회의)

- 1월 28일(토), 2월 11일(토) 오후 세시
- 부산 서면 → '컨텐츠코리아랩' 등에서 진행

비대면 회의

- 평일 저녁 **주3회** 진행(월, 화, 목 / 오후 8-9시)
- ZEP에서 진행

📌 코어타임

- 평일 저녁 **주3회** 미팅 이후(월, 화, 목/ 오후 9-12시)

📌 티타임

- 회의시간, 코어타임 사이 10-20분씩 가지기

📌 의사결정 방식

- 애자일 방식

📌 불만을 이야기하는 채널

- 노션 페이지 활용
 - 불만 사항을 대략적으로 해당 노션 페이지에 작성해두면 회의시간에 확인해 회의 내용으로 다룰 예정

📌 지양해야 할 것

- 회의 불참 시 사유 미리 이야기 하기
- 어떤 아이디어를 제시하든 비난하지 않기
- 적극적으로 참여하기

👤 기타 사항

✓ 대면미팅 시 취미 공유시간 갖기

- 게임, 볼링 등 취미를 공유하면서 돈독한 팀워크 형성하는 시간을 갖기로 함

Development Plan

개발계획

JAN

FEB

	1	2	3	4	1	2	3	4
문서	PMP작성 자료조사 아키텍처 설계		erd 설계 API 명세서 작성 와이어프레임 제작					
주요 기능			화면 구성 뷰 구현 회원 관리 기능, 영상 공유, 채팅 기능 구현		FE, BE 완성 기능 순 연동			
테스트						성능 테스트		

Front-End

- **개발 환경 세팅 및 컴포넌트 설계 (-01.15)**
 - 1) 프레임워크, 라이브러리 등 개발 스킬 선택
 - 2) 체계적인 컴포넌트 제작을 위한 화면 컴포넌트 분석 및 분리
- **와이어프레임 제작 (01.15-01.30)**
 - 1) Figma를 통한 구체적인 화면 와이어프레임 제작
- **작은 단위의 컴포넌트부터 제작 (01.15-01.30)**
 - 1) 작은 단위의 컴포넌트에서 페이지 단위로 화면 뷰 및 프론트단 기능 구현
- **api 연동(02.01-02 중순)**
 - 1) 백엔드와 기능별 연동 진행 및 기능 테스트

Back-End

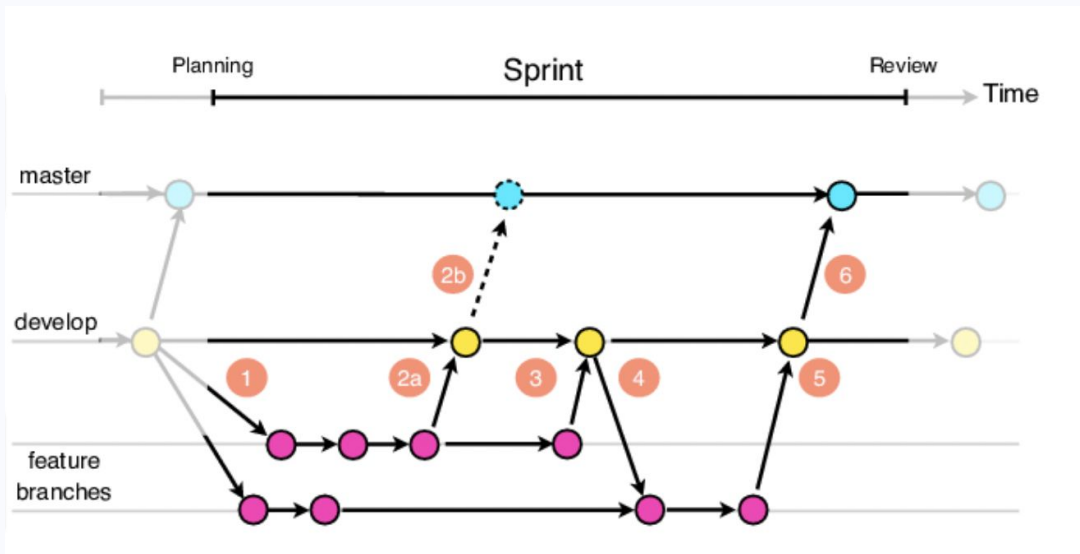
- **개발 환경 구축(01.15-01.18)**
 - 1) 도커 환경 세팅
 - 2) 서버 구성 설계

- **영상플랫폼 기본기능 구현 (01.19-01.25)**
 - 1) 화면 공유 서버
 - 2) 사용자관리 서버

- **영상 공유 기능 구현 (01.26-02.04)**
 - 1) 실시간 통신 기능 구현
 - 2) 영상 공유 기능 테스트

- **채팅 서비스 (02.05 - 02.11)**
 - 1) 채팅 도메인 구성
 - 2) 유저간 채팅 서비스
 - 3) 전체 기능 테스트

Git Flow



master

- 제품으로 출시될 브랜치

develop

- 개발이 완료된 최신 브랜치
- 신규 개발된 내역이 처음 합쳐지는 브랜치

feature

- 각 기능을 개발하는 브랜치

Git Convention

Git Message Structure

- 기본적으로 두 줄을 기본으로 한다.

Commit Type

- Feat : 새로운 기능 추가
- Fix : 버그 수정
- Docs: 문서 수정
- refactor : 코드 리팩토링
- test : 테스트 코드
- style : 코드 변경이 없는 경우
(ex. 세미콜론 추가 정도)

Subject

- 50자를 넘기지 않는다.
- 대문자로 작성한다.
- 마침표를 붙이지 않는다.

[깃 메뉴얼 문서 작성]

- PR 보내는 방식
- PR rebase 방식
- PR Merge 방식
- Commit 방식
- 각 브랜치 관리 방법
- 주의할 점
- 예제

[브랜치 종류] - origin

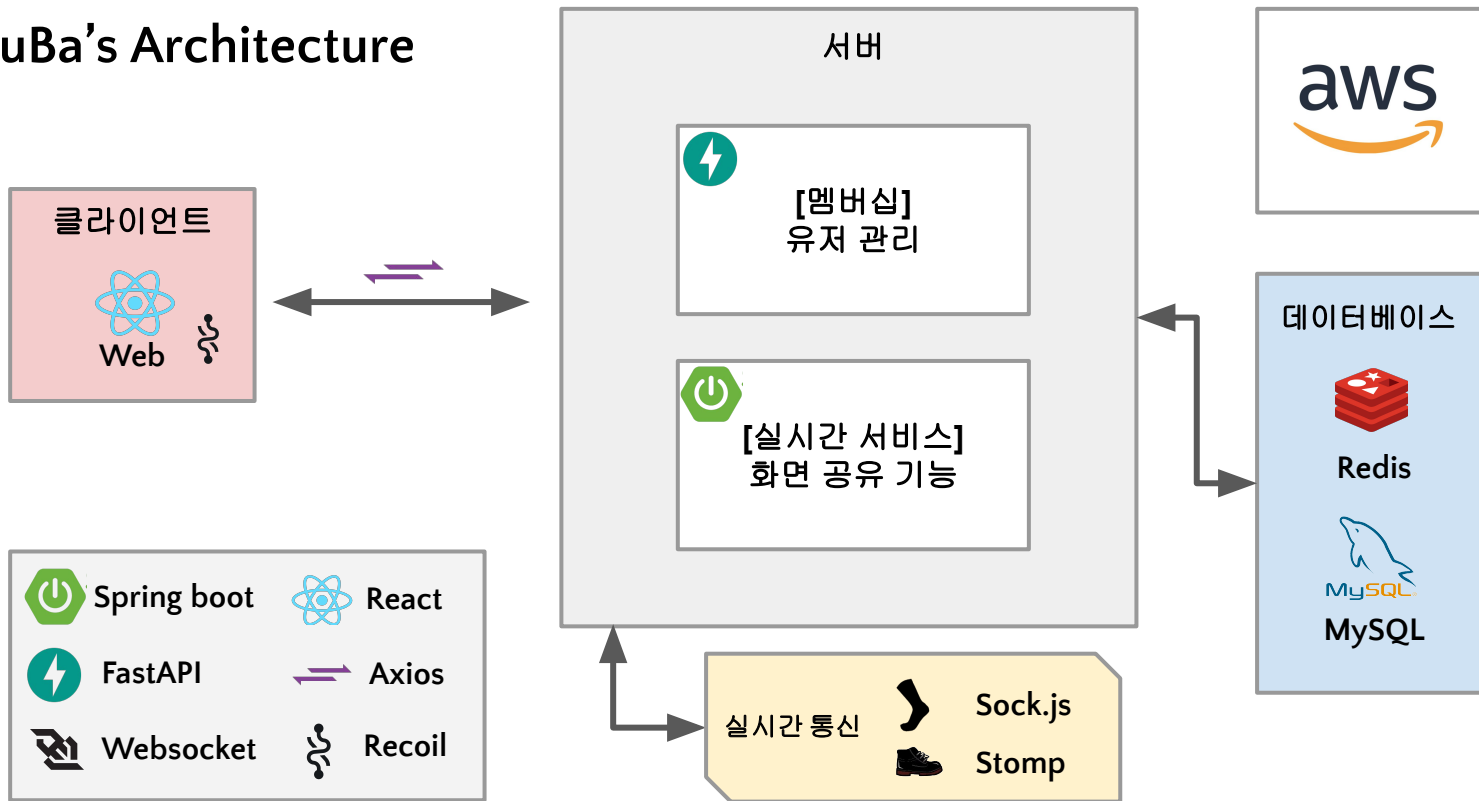
- 1) master : 제품으로 출시될 브랜치
- 2) develop
 - 개발이 완료된 최신 브랜치,
 - 신규 개발된 내역이 처음 합쳐지는 브랜치
- 3) feature : 각자의 개인 작업 브랜치

[브랜치 종류] - local

- 1) main
- 2) PR 용도 브랜치
- 3) 작업 브랜치



eGuBa's Architecture



Frontend

페이지 목록	라이브러리, 오픈소스	주요 기능
로그인 /회원가입 /회원정보 수정	-	<ul style="list-style-type: none">- ID, Password- 이름, 아이디, 아이디 중복 확인, 비밀번호, 비밀번호 확인, 이메일, 이메일 인증, 생년월일- 닉네임, 비밀번호, 이메일 연동 정보 변경(아이디 x), 회원탈퇴
원격 영상 공유 페이지	SockJS	<ul style="list-style-type: none">- 채팅- 영상 플레이어
동영상 플레이어	react-player	<ul style="list-style-type: none">- 동영상 공유
영상 공유 관련 모달창	SockJS	<ul style="list-style-type: none">- 영상 공유 버튼- 사용자간 채팅(추가기능)

Backend - Server

서버	프레임워크 및 라이브러리	상세 설명	기능
영상 공유	Stomp Redis	영상 싱크 동작	웹 소켓 영상 스트리밍 Redis: 공유 url 저장
채팅	Stomp Redis	실시간 메시지 전송	채팅 중계 기능 Redis: 방정보, 접속유저 정보 저장
멤버십	FastAPI MySQL Redis	유저 서비스	유저, 관리자 데이터 수집 및 관리 저장 데이터에 대한 CRUD 기능

Backend - Server (DB 아키텍처)

RDBMS : MySql

User			
🔑	user_type	Domain	String(20)
🔑	user_id	Domain	String(40)
	encrypted_password	Domain	String(100)
	nick_name	Domain	String(100)
	email	Domain	String(40)
	user_status	Domain	Integer

In-Memory : Redis

저장 데이터	저장 형태
방 링크, 호스트 이름	Set 자료구조 < String, String >
방 링크, 사용자 정보	Set 자료구조 < String, String >
방 링크, 영상 Url	Set 자료구조 < String, String >
이메일 인증정보	Set 자료구조 < String, String >