
Member

김종준



임채승



이하얀



정하영



Goals - Team

더 좋은 개발자로 성장하기

기존의 코드에서 벗어난다, 유연한 설계, 기술 선택 이유를 설명할 수 있는 상태

1. 평소 고려하지 않았던 대량의 트래픽이라는 문제를 해결해나가며 더 좋은 개발자로 성장하자
사용해 본 적 없는 기술(프로토콜, 아키텍쳐)을 이용해서 문제를 해결한다.
그 외 문제는 팀원과 상의하여 문제를 해결하기 위한 기술을 선택한다.
2. 새로운 기술을 도입 시 팀 리뷰와 문서화를 통해 더 좋은 개발자로 성장하자

Project Introduction

대량의 트래픽을 받는 프로젝트

.now를 기반으로 하는

Live
Now.



Goals - 임채승

Current

Socket 통신 하는 법을 안다

After

Socket 통신을 활용한다

Web에서 HTTP의 정확한 규격을 모른다

Web에서 HTTP 헤더를 변경하여 규격에 맞게 통신할 수 있다

React같은 라이브러리를 잘 모른다

React의 생명주기를 안다

Goals - 임채승

문제를 해결하기 위해서 다양한 방법으로 시도하는 개발자 되기

1. 라이브러리의 사용 이유 및 컨셉을 알기
2. 문서를 통한 것이 아닌, 직접 시도해보기

How To Work

1. React의 생명주기를 공부
2. HTTP헤더에 대해 공부
3. Socket의 활용에 대해 찾기

Goals - 정하영

Current

대용량 서비스를 고려해서 설계 해본 경험이 없다.

After

대용량 서비스를 고려한 **설계 능력**을 갖춘 상태

왜 해당 기술을 사용할지 고민이나 이유 없이 무작정 도입했다.

기술 선택 이유 설명할 수 있는 상태

Goals - 정하영

더 나은 백엔드 개발자로 성장하자

1. 대용량 서비스를 고려한 설계 및 개발 능력
2. 클린 코드 작성 능력 함양된 상태

How To Work

1. 개발 기간 동안 진행

기능 구현 후 정적 코드 분석툴을 이용하여 코드 중복, 코드 스멜 등을 측정하고 global maintainability rating을 A등급으로 유지한다.
아키텍처 설계 및 기능 구현 시 병목 지점을 고려한다.

2. 프로젝트 완료 시점 진행

기능 구현 후 병목지점으로 예상되었던 api들에 대해 성능테스트를 진행하여 서버 성능을 개선시키며 일련의 과정은 문서화한다.

Goals - 이하얀

Current

서버 개발의 효율성과 생산성이 부족한 상태

After

서버 개발 시에 **효율성과 생산성을 고려할 수 있는 상태**

큰 규모의 서버에서 발생할 수 있는 서버 성능에 따른 부하 정도를 고려하지 않고 서버를 개발하는 상태

큰 규모의 서버에서 발생할 수 있는 서버 성능에 따른 **부하 정도를 고려하며 서버를 개발할 수 있는 상태**

Goals - 이하얀

무작정 코드만 쓰는 것에서 벗어나 개발의 효율과 생산성을 고려하고, 코드 외적인 부분까지 생각하며 진짜 '개발'을 하는 백엔드 웹 개발자 되기

How To Work

자세한 내용은 부록에

1. 프레임워크 학습 및 적용
2. 서버의 다양한 성능 개선 방식 파악
3. 성능 개선 방식에 따른 서버 동작, 처리 과정의 차이 이해 및 명세
4. 성능 테스트 적용을 위한 연습
5. 서버에 성능 테스트 적용

Goals - 김종준

Current

Text 기반의 REST API 개발 경험만 있음

After

Media 서비스를 이해하고 오픈소스를 활용하여 개발 할 수 있는 상태로 성장

대량의 사용자를 생각하고 개발을 진행한 경험이 없음

대량의 사용자를 염두한 다양한 아키텍쳐를 고려할 수 있는 개발을 할 수 있는 상태로 성장

Goals - 김종준

1. Media

스트리밍 서비스 개발로 RTMP to HLS의 스트리밍 서비스 개발을 목표
단일 화질이 아닌 다수의 화질 제공을 목표

2. 대량의 트래픽

Now와 같이 갑자기 치솟는 트래픽을 어떻게 감당할 것인지 이에 맞는 다양한 아키텍쳐 구조를 경험해 보고 각 아키텍쳐마다 장단점을 파악할 수 있는 지표 획득을 목표

How To Work

1. Media

Media 서버의 경우 오픈 소스를 활용하여 미디어 서버 개발
Media 서버를 제어할 수 있는 메니저 서버 개발

2. 대량의 트래픽

아키텍쳐 조사

Planning

임채승

스트리밍 서비스 분석 ~1/10
채팅 및 영상 송출 방법 파악 ~ 1/15
화면 구성 ~ 1/16
API연결 ~ 2/2
FE 기능 테스트 ~ 2/18

정하영

대용량 시스템 설계 기초 서적과 구글링을 통해 채팅 도메인 배경 지식 학습, 개략적 아키텍처 설계 ~1/11
아키텍처 리뷰 후 피드백 반영 및 구체화 ~1/14
아키텍처 기술 관련 학습 ~1/19
채팅 구현 ~1/26 신고 구현 ~2/2
성능 테스트 및 개선 ~ 2/18

김종준

Spring cloud 관련 공부 ~ 1/5
유저, API Gateway 구현 ~ 1/12
유저, API Gateway 테스트 및 개선 ~ 1/19
WebRTC관련 공부 ~ 1/19
화상 통화 구현 ~2/2
화상 통화 성능 테스트 및 개선~ 2/18

이하얀

Spring 강의를 통한 프레임워크 학습 ~1/13
인증(로그인, 회원가입 등), 유저 서버 아키텍쳐 설계 및 수정 ~1/11
성능 테스트 관련 공부 ~1/15
유저 서버 구현 ~ 1/29 그외 자잘한 부가 기능 구현 ~ 2/5
성능 테스트 및 개선 ~ 2/18

Cooperative Work

FrontEnd - Backend

API - Swagger or Postman

Backend

코드 올리는 방법

개인 브랜치에서 PR → Approve → Rebase

브랜치 전략

JIRA에서 ID값을 가지고 와서 관리

Benefit Management

협업

스크럼 월-금 14:00, 최대 10분

전날 자신이 하지 못 한일, 하지 못한일이 있다면 왜 그렇다고 생각하는지, 오늘은 무엇을 할 것인지 등을 공유하는 시간
이점: 우리팀이 무엇을 하는지 알 수 있다.

정기회의 목 14:00 - 16:00

목요일마다 회의를 통해서 백로그 관리, 저번주 스크린트 달성을 확인하여 이번주 스프린트 작업 설정
순서를 정하여 서기, 발표자 역할을 수행
이점: 팀원들과 소통하는 능력을 향상시킬 수 있다.

Risk Manageing

1. 프로젝트 요구사항 변경 또는 일정 차질

회의시간과 스크럼 시간에 이야기 한다.

2. 기능의 의존성에 대한 관리

CPM문서를 작성하여 서로 관리한다.

3. Git 충돌

브렌치 전략을 사용한다.

Thank you

How To Work(detail) - 임채승

1. React의 생명주기를 공부

React의 사용법이 아닌 컨셉에 대해서 공부(Learning React)

개인 Notion에 문서화

2. HTTP헤더에 대해 공부

HTTP헤더가 어떤 역할을 하는지 공부(Googling)

개인 Notion에 문서화

3. Socket의 활용에 대해 찾기

Socket의 통신

SockJS에 대한 컨셉 조사

개인 Notion에 문서화

How To Work(detail) - 이하얀

1. 프레임워크 학습 및 적용

Spring 중 팀프로젝트 개발에 필요한 핵심 기술을 강의로 빠르게 학습
개인 Notion에 문서화

2. 서버의 다양한 성능 개선 방식 파악

다양한 방법 사용(공식 문서, 구글링, 관련 서적 등)
개인 Notion에 문서화

3. 성능 개선 방식에 따른 서버 동작, 처리 과정의 차이 이해 및 명세

서버에 최선인 성능 개선 방식에 대한 고민
개인 Notion에 문서화

How To Work(detail) - 이하얀

4. 성능 테스트 적용을 위한 연습

개발하고자 하는 서버의 '일반적인' 부하 테스트 시나리오 작성 연습

성능 테스트 툴(nGrinder 등)의 사용

시나리오에 의한 부하 테스트 진행(User, TPS, Time 등의 지표 활용, 테스트 타겟 시스템의 범위 등 고려)

개인 Notion에 문서화

5. 서버에 성능 테스트 적용

적용할 각 서버 성격에 맞는 성능 테스트 시나리오를 설계 및 작성해 부하 테스트 진행

다양한 시나리오를 작성해보고, 팀원들과 협의 과정을 반드시 거쳐 각자 개발한 서버의 모든 범위에서 문제가 없는지 검증을 거칠것.

각 서버의 성격에 맞는 성능 테스트 시나리오를 직접 작성하여 부하 테스트 진행