JIKJI팀 PMP 문서

김정현 박종하 이한솔 김영현

목차

캠프 참여 동기

팀 목표

주제 및 선정 이유

주제에 따른 개인목표

workflow

개발 계획

Architecture

API 명세

JIKJI

캠프 참여 동기

김정현	• 개발자로서 스스로 무엇이 부족한지 알고 싶다. • 외부 프로젝트를 통해 협업을 어떤식으로 진행하는지 배우고 싶다. • 어떻게 노력해야 꾸준히 성장하는 개발자가 될 수 있는지 알고 싶다.
박종하	• 백엔드 개발자로서 서비스 개발에 참여해보고 싶다. • 전문가 분들의 피드백을 통해 개발 프로세스를 체계적으로 경험해보고 싶다. • 가치있는 코드를 짜는 방법, 개발자로서의 태도, 방향성을 알고 싶다.
이한솔	• 이전 프로젝트에서 정리가 안된 느낌을 받았다. • 현업 개발자에게 멘토링, 코드 리뷰를 받고 체계적인 프로세스를 통해 완성도 높은 팀 프로젝트를 경험하고 싶다.
김영현	• 팀 프로젝트를 통해 규모 있는 서비스를 만들어보고 싶다. • 협업 과정에서 발생할 수 있는 문제점들을 해결하는 과정에서 현업에서의 문제해결 방법을 경험해보고 싶다.

팀 목표

전문적인 현업 개발 프로세스를 통해 완성도 높은 팀 프로젝트 경험

=> 지속 성장 가능한 개발자 되기

+

체계적인 협업 경험을 통해 소프트 스킬 습득

주제 및 선정 이유

Instagram

선정 이유

- 다양한 서비스들이 혼합되어 있어 MSA 구조로 개발하기 유리
- 심플한 UI => 백엔드 개발에 집중
- 참고 자료가 많아서 개발이 용이
- 대용량 트래픽을 가진 서비스의 서버 아키텍쳐를 구현해보면서 대형 서비스에서 서버 아키텍쳐를 선정, 구현하는 원리를 알아보고 싶음
- 여러 서비스들을 함께 개발하면서 협업에 대한 이해도 증진 및 소프트 스킬 습득 가능

개인 목표 – 김정현

세부 목표

1. 프로젝트에서 유지보수에 적합한 코드를 짜는 방법, 성능 최적화에 대해 공부하고 구현하는 경험을 통해 개발자가 어떤 방식으로 공부하고 서비스를 발전시켜야 하는지 배우고 체화하기
2. MSA 아키텍쳐를 구현하면서 대형 프로젝트에서 서버 아키텍쳐를 선정, 구현하는 원리를 배우고 MSA 아키텍쳐를 합리적으로 설계할 수 있는 능력 가지기
3. 개발 전 사전 협의, 스크럼 회의, git을 이용한 버전 관리 및 문서 관리를 체계적으로 경험하고 동료와 함께 개발할 수 있는 개발자 되기

프로젝트 후 변화 : 방법을 찾아내는 개발자, 함께 하고 싶은 개발자 되기
=> 문제 해결 방법을 찾아가는 나만의 루틴을 찾기 (수정)

- 지금까지는 오픈소스 활용 등을 통해 구현 외에 다른 요소를 고려하지 않는 개발을 해왔는데, 프로젝트를 통해 유지보수와 성능, 협업을 고려해 개발할 수 있는 개발자가 될 것이다.
- 대형 서비스를 개발해본 적이 아직 없다. 이번 프로젝트를 통해 MSA 아키텍쳐를 설계, 구현하면서 대형 서비스를 설계, 구현하는 접근 방식을 배우고 스스로 할 줄 아는 개발자가 될 것이다.

개인 목표 – 김정현

1. 프로젝트에서 유지보수에 적합한 코드를 짜는 방법, 성능 최적화에 대해 공부하고 구현하는 경험을 통해 개발자가 어떤 방식으로 공부하고 서비스를 발전시켜야 하는지에 대한 느낌 잡기

유지보수에 적합하고 성능이 뛰어난 코드를 짜는 방법 익히기

- 리팩토링, 클린코드 도서를 참고해 디렉토리 분류 및 코딩 규칙 만들고 지키기
- 코드 리뷰, 검색을 통해 성능에 대해서 오랫동안 고민해보고, TPS 측정 등의 성능 테스트, 로그 분석 등을 통해 성능 관점에서 개발하는 방법 익히기

개발자의 공부 방법, 사고 방식 배우기

- 멘토님들과 윈터데브캠프 참가자 분들께서 공유해주시는 개발 관련 글, 영상, 도서를 보고 개발자는 어떤 방식으로 생각하고 개발해야 하는지를 나름대로 이해해서 체화시키기
  => 스스로 만족할 수 있는 코드를 짤 수 있는 개발자로 발전하기

개인 목표 – 김정현

2. MSA 아키텍처를 설계, 구현하면서 대형 프로젝트에서 서버 아키텍처를 선정, 구현하는 원리를 배우고 MSA 아키텍처를 합리적으로 설계할 수 있는 능력 가지기

MSA 아키텍처 설계할 수 있는 개발자 되기

- MSA를 사용하는 구체적인 사례를 찾아서 서비스를 분리하고 통합하는 기준과 아키텍처를 구성할 때 고려할 점 등을 배워 MSA 아키텍처를 프로젝트에 자연스럽게 사용할 수 있게 되기

MSA 아키텍처 구현할 수 있는 개발자 되기

- MSA 형태로 서비스를 개발하면서 발생하는 트랜잭션 처리 문제, 서비스간 의존성 문제 등을 겪고 해결하면서 MSA로 설계한 서비스를 구현하는 과정을 습득하고 과정을 문서화하기

개인 목표 – 김정현

3. 개발 전 사전 협의, 스크럼 회의, git을 이용한 버전 관리 및 문서 관리를 체계적으로 경험하고 동료와 함께 개발할 수 있는 개발자 되기

버전 관리

- 브랜치, 히스토리 관리 등 Git에서 버전 관리에 필요한 개념과 활용에 익숙해지기
- 협업에 적합한 Git 관리 방법을 익히기
  => Git에 대한 두려움 없애기 !

문서 관리

- API 명세서 작성에 익숙해지기
- 이슈 발생시 어떤 방식으로 해결했는지 따로 기록하고 정리하기
  => 협업시에도 개발 역량을 100% 활용할 수 있는 개발자 되기

개인 목표 – 박종하

세부 목표

1. 스프링 프레임 워크를 바탕으로 한 서비스 개발을 해보며, 백엔드 구현 능력 향상
2. 개발 과정 작성 및 이슈 해결 내용, API 기능 명세서 문서화 작업하여 개발 작업 기록, 협업을 통하여, 개발 과정에서 협업 능력 및 의사 소통 능력 쌓는 체계적인 개발 프로세스 체험.
3. msa 아키텍쳐 이해 및 대규모 서비스 msa 를 구현

프로젝트 후 변화 : 천천히 성장하는 꼼꼼한 개발자 되기.

- 백엔드 서비스 프레임워크에 어느 정도 익숙해진 후, 주먹구구식 코드가 아닌 지속 가능한 코드를 짜는 법에 대해 고민할 줄 아는 백엔드 개발자 되기
- 체계적인 개발 프로세스를 경험한 후, 앞으로의 프로젝트에서도 다음과 같은 방법론을 적용하여 개발과정을 기록하고 문서화 하며, 팀원들과 협업할 수 있는 개발자 되기.
- msa 을 구현할 줄 알며 대규모 프로젝트 서비스에서 msa 의 중요성 깨닫기

개인 목표 – 박종하

1. 스프링 프레임 워크를 바탕으로 한 서비스 개발을 해보며, 백엔드 구현 능력 향상

Spring boot 와 JPA 를 활용하여 웹 서버 개발

- Spring boot 작동원리와 기본 개념에 관해 공부 및 JPA 활용능력 쌓기
- Spring mvc 패턴을 이해하고 구현하기
- REST API 설계 및 개발 능력 갖추기
- 이론으로 배웠던 데이터 베이스 실전 적용하며 체감하기.
-

클린 코드 및 테스트 코드를 고려하는 개발자 되기

- 추 후 프로젝트가 끝난 후, 기능 개발이 완료되면 코드를 리팩토링하며 클린 코드 원칙에 따라 작성했는지 살펴 볼 줄 아는 개발자 되기.
- 또한, 테스트 코드의 필요성을 이해하고 작성해 보기

개인 목표 – 박종하

2. 개발 과정 작성 및 이슈 해결 내용, API 기능 명세서 문서화 작업하여 개발 작업 기록, 협업을 통하여, 개발 과정에서 협업 능력 및 의사 소통 능력 쌓는 체계적인 개발 프로세스 체험.

개발 과정을 꼼꼼히 기록

- 이슈 내용과 해결 방법을 기록. 문제점 및 해결과정 그리고 도출점 등을 개발 일지에 정리.
- 구현한 API 들을 기능 명세서를 활용하여 정리.
- Entity 등을 diagram과 use case 등을 활용하여 개발 과정에서의 flow 작성하여 시각화.

협업을 통한 소프트 스킬 능력 향상

- 구현 과정에서의 문제점들을 개발 일지에 작성 및 팀원들과 공유.
- 팀원들과 매일 스크럼 회의, 그리고 매 주 개발 일지를 작성하며 개발 과정을 공유하며 의사 소통 능력 증진.

개인 목표 – 박종하

3. 대규모 서비스를 분석하고 msa 아키텍쳐 이해 및 인스타 그램의 msa 를 구현해보기

분석 및 이해 단계

- msa 아키텍쳐 이해 및 예제 찾아보기
- 인스타 그램의 서비스 분석과 그에 필요한 MSA 서버 나누는 기준 정립.
- 서버 간 통신 방법과 그에 필요한 프레임 워크 및 오픈 소스 조사.
- 구현할 서비스와 관련된 아키텍쳐 조사

설계 단계

- 분석 및 이해한 내용을 바탕으로 구현할 서버 아키텍쳐 및 디비 설계
- 아키텍쳐를 설계 할 때 각각을 선택한 기준과 합당한 근거를 바탕으로 설계 하기. 이에 대해서 팀원들과 논의 및 멘토님께 중간 리뷰 받기.

구현 단계 및 검증 단계

- 분배한 서버 아키텍쳐 및 api 구현하기. 이에 대해 api 명세서와 아키텍쳐 명세서 선 작성
- 과정에서의 문제점들을 개발일지에 작성. 이슈 내용과 해결 방법을 기록

개인 목표 – 이한솔

세부 목표

1. MSA를 직접 구현하고 배포하며 서로 통신하는 방법에 대해 이해한다.
2. REST API와 클린코드를 지키며 코드를 작성한다.
3. 개발하며 발생한 이슈와 고민은 나중에 보더라도 이해할 수 있도록 항상 정리한다.

프로젝트 후 변화 - 변경에 흔들리지 않는 개발자

1. 명세를 꼼꼼히 하는 개발자
2. 회고하는 개발자
3. 개발하면서 항상 WHY를 생각하는 개발자

개인 목표 – 이한솔

1. MSA를 직접 구현하고 배포하며 서로 통신하는 방법에 대해 이해한다.
    - 마이크로서비스 분리 단위, 사용하지 말아야할 경우 비교
    - 엔터프라이즈급 아키텍처 분석해보고 이해하기
    - MSA로 설계한 시스템을 배포하는 방법 이해
    - 마이크로서비스로 분리된 구조에서 데이터의 일관성을 지킬 수 있는 방법 이해
    - 마이크로서비스간 통신 프로토콜 gRPC or HTTP 조사
2. API 구현 및 클린코드 작성 원칙 지키기
    - API 문서에 리소스를 접근하기 위한 적절한 request 정보 생각하기
    - 각 메서드는 한가지 역할만 담당하도록, 작게 만들도록 노력
        - 각 PR마다 코드리뷰를 필수적으로 받아 피드백 받기
    - 나중에 구조가 변경되더라도 같은 결과를 내는지 예외 상황을 다룰 수 있도록 확인하기 위한 기능 테스트 작성
3. 개발하면서 발생한 이슈, 고민은 나중에 보더라도 이해할 수 있도록 정리해두기
    - 왜 적용했는지, 고민을 하게 된 계기부터 해결 과정 그리고 결과 값으로 정리

개인 목표 – 김영현

세부 목표

1. git 버전관리를 학습하여 팀 프로젝트에서 git 관리 실패하지 않기
2. MSA 직접 설계와 구현을 하며 MSA의 철학 이해하기
3. 기능 명세와 개발 과정, 문제해결 내용을 정리하여 명세서 작성하기
4. 프론트엔드를 React로 설계하여 백엔드와 프론트엔드의 협업 방식 이해하기

프로젝트 후 변화

대외 활동, 회사 서류전형 프리패스하는 개발자 되기

1. 프로젝트 진행 시 기능명세서, 개발일지 정리를 통해 개발을 일목요연하게 진행하는 개발자
2. git 버전관리를 통한 프로젝트 진행상황을 정리하는 개발자
3. 프론트엔드와 협업 시 기능 조율과 명확한 의사전달을 하는 개발자

개인 목표 – 김영현

1. git 버전관리를 학습하여 팀 프로젝트에서 git 관리 실패하지 않기

- Git 관리 기본 강의와 캠프에서 제공한 git 협업 실무 강의를 통한 Git 버전 관리 기본기 획득
  - GUI 강의를 통해 GUI를 활용한 git관리 학습
  - GUI의 능숙도가 향상되면 CLI를 활용한 git관리로 전환
- Git 커밋 메시지 컨벤션 학습을 통한 협업 과정에서 버전 변경사항 전달력 향상
- Git 이슈 관리 학습을 통한 협업 과정에서 이슈 발생 시 이슈 해결능력 향상
  - Github Issue Template 사용법을 학습하여 이슈 상황 발생 시 정리
- Git 브랜치 관리와 협업 중 브랜치 병합 과정과 충돌 해결능력 향상
  - 브랜치 병합 방법 학습과 브랜치 충돌 시 주의사항 학습

개인 목표 – 김영현

2. MSA 직접 설계와 구현을 하며 MSA의 철학 이해하기

- 서버간 통신 방법, 프레임워크, 도구 사용 방법, 기업들의 MSA 도입 이유 조사하기
  - 조사한 내용 정리 요약 후 이해하기
- MSA 도입시 발생할 수 있는 장애들과 이를 해결하기 위한 방법 찾아보기
  - MSA를 도입하지 않은 기업들은 왜 도입을 하지 않았는지 찾아보기
- 조사한 내용을 바탕으로 서버 아키텍처를 그리기
  - 서버 아키텍처 작성 후 팀원들과 회의를 통해 서버 수정, 삭제, 추가하기
  - 아키텍처 리뷰 시간에 조사한 내용과 사용 도구, 아키텍처에 대해 캠프장님과 멘토님께 피드백 받기
- MSA 적용하기
  - 각 서버별 API 명세서 작성 후 팀원들과 API명세서 수정부분에 대해 토의하기
  - 문제 발생 시 이슈 정리하여 멘토님께 질문하기

개인 목표 – 김영현

3. 기능 명세와 개발 과정, 문제해결 내용을 정리하여 명세서 작성하기

- Swagger, Rest docs를 활용하여 API 명세서 작성
- Github Issue를 활용하여 이슈와 해결과정 정리
  - 원인 분석 후 이슈 정리
  - 하루정도 이슈 처리 후 해결안될 시 멘토님께 질문
- 개발 진행 중 설계가 아닌 설계 후 개발 진행
- 개발 변경 사항 발생 시 노션 활용하여 정리
  - 팀원들과 변경 사항 공유
- 개발일지를 매일 작성하여 개발 과정 정리

개인 목표 – 김영현

4. React로 프론트엔드를 설계하여 백엔드와 프론트엔드의 협업 방식 이해하기

- React 라이브러리 기본 강의를 수강하며 기초 지식 쌓기
  - 노마드코더 기본 React 실습 강의 수강하여 기본기 쌓기
  - React 공식 문서를 참고하여 백엔드와 프론트엔드 연결하기
- 프론트엔드의 역할과 개념 조사하기
  - 백엔드에서 왜 프론트엔드가 파생되었는지 학습하기
  - 백엔드와 프론트엔드가 협업 시 발생할 수 있는 문제상황과 주의해야 할 점 조사하기

Workflow

일일 스크럼 회의

시간 : 매일 아침 10시

장소 : 슬랙 허들

회의 안건 :

- 전날 스크럼 회의에서 세운 목표사항 완료 여부 체크
- 새로운 일일 목표 공유
- 문제 상황 공유

주간 스프린트 회의

시간 : 금요일 저녁 8시

장소 : 슬랙 허들

회의 안건 :

- 스프린트 진행 상황 점검
- 마일스톤 환기
- 문제 상황 공유

Workflow

팀빌딩

주기적으로 팀의 목표와 개인의 목표를 되돌아보는 시간 가지기

협업을 위해 필요할 경우 회의를 주최하고 되도록 참석하기

개발 상황에 대해 솔직하게 이야기하고 서로 신뢰하기

그라운드룰

회의-슬랙, 노션-자료공유

만장일치로 의사 결정하기

네이버 자바 코딩 컨벤션 활용

결석 많이 하지 않기!

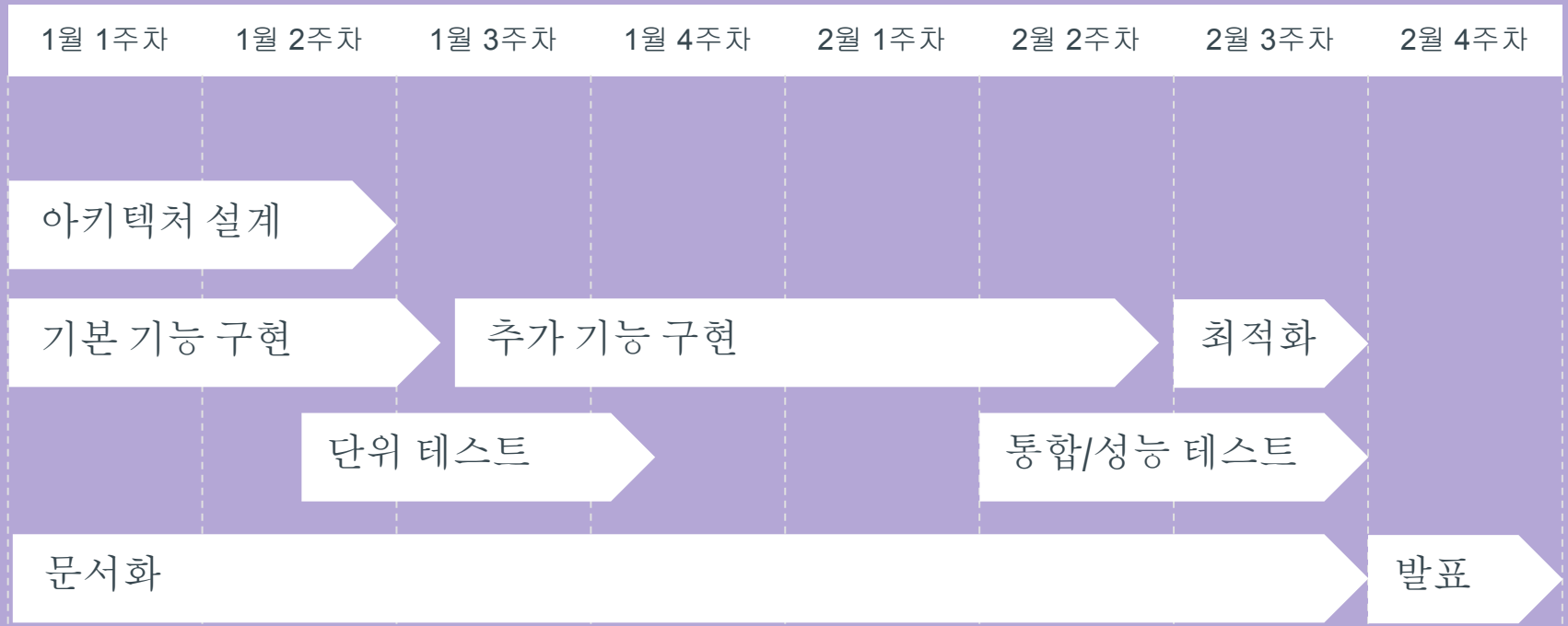
리뷰시 서로 존중하기

개발 계획

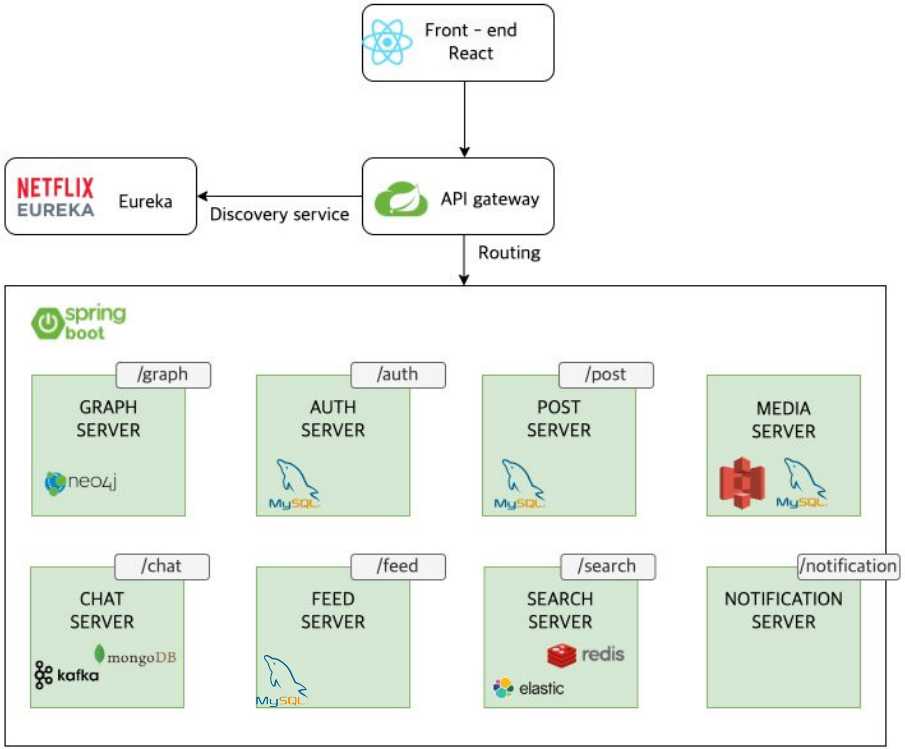
Milestone

1. SNS 기본 기능 구현 - Auth, Post, Feed, Media, Graph(follower) 서버
2. 추가 기능 구현 - Notification, Search, Chat, Story 서버
3. 테스트 및 성능 최적화

개발 계획


1월 1주차
1월 2주차
1월 3주차
1월 4주차
2월 1주차
2월 2주차
2월 3주차
2월 4주차
아키텍처 설계
기본 기능 구현
추가 기능 구현
최적화
단위 테스트
통합/성능 테스트
문서화
발표

Architecture



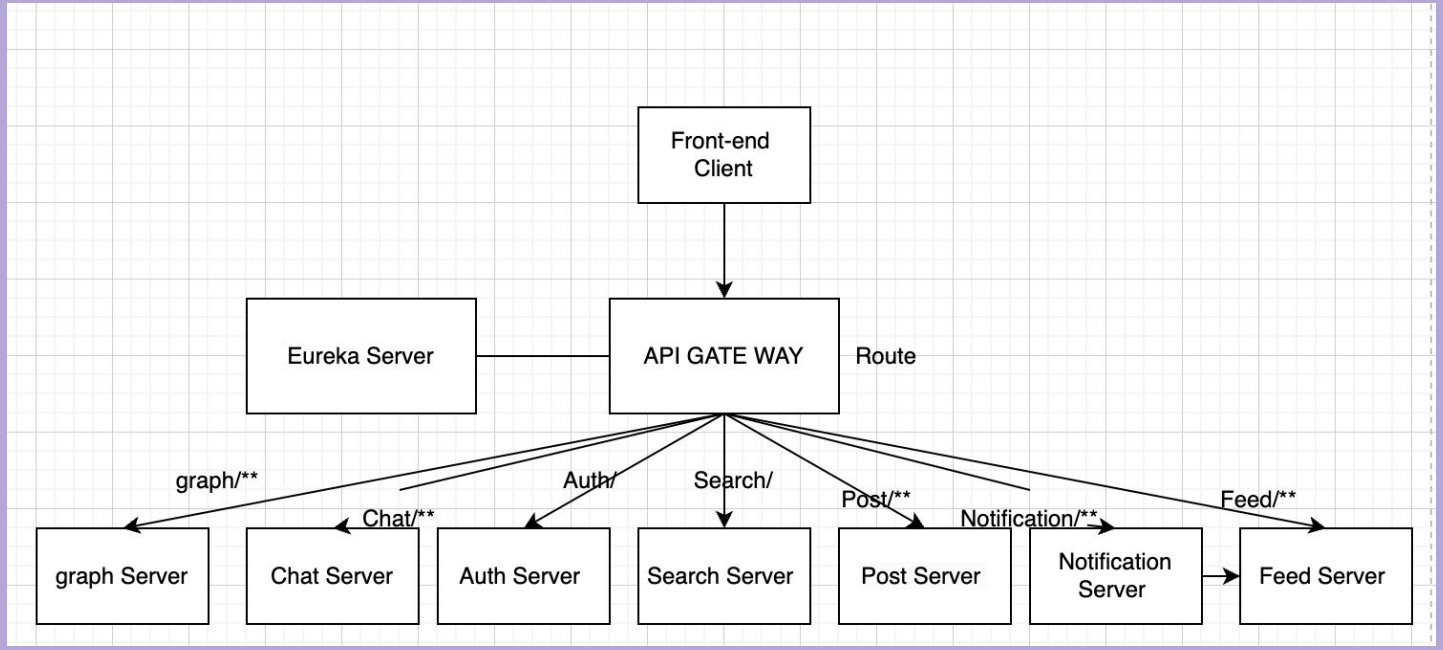
feed : mysql , Springboot
Post & comment : mysql , Redis, Springboot
Media : mysql, AWS S3, Springboot

Graph : springboot, neo4j
Notification : springboot, mysql

Search : springboot, elasticsearch

Auth : MySQL, Springboot
Notification : springboot
Chat : springboot, nosql, websocket

Gateway Server



API 명세

AUTH Server

홈화면

Aa 이름	Method	URI	참고	request	response
홈 화면 가져오기	GET	/			
로그인 실행하기	POST	/user/login	로그인 시 accessToken, refreshToken 발급		JWT(accessToken, refreshToken)
회원가입 페이지 이동하기	GET	/signup			
비밀번호 찾기 페이지 이동하기	GET	/searchPassword			

회원가입

Aa 이름	Method	URI	참고
회원가입 실행하기	POST	/signup/enroll	회원가입 후 메일인증 실행
메일 인증 페이지 이동하기	GET	/signup/mail	
메일 인증 실행하기	POST	/signup/mailAuth	메일인증 실패 시 회원 기록 삭제

API 명세

AUTH Server

비밀번호 찾기

Aa 이름	태그	URI	참고
회원 조회	POST	/searchPassword/authUser	회원 내역 조회 성공 시 임시비밀번호 전송

JWT 검증

Aa 이름	태그	URI	참고	request	response
accessToken 검증	POST	/accessToken/validateAccessToken		accessToken	true, false
refreshToken 검증	POST	/refreshToken/validateRefreshToken	인증 성공 시 accessToken 발급	RefreshToken	Map
refreshToken 삭제	POST	/refreshToken/deleteRefreshToken		userEmail	NotFoundUserEmail, SUCCESS
User Roles 발급	POST	/accessToken/getRoles	User, verifiedUser 구분 accessToken 검증 후 발급 요청	accessToken	User, verifiedUser

API 명세

Feed Server

Feed API

Aa 이름	태그	description	response
/feed/{user_name}	GET	user_name 에 대한 feed 가져옴	user_id, post_id, username, createdAt

API 명세

Post & Comment Server

Posts

표

Aa 이름	Method	URI
모든 포스트 가져오기	GET	/posts/{user_id}
포스트 작성하기	POST	/posts
포스트 수정하기	PATCH	/posts/{post_id}
포스트 삭제하기	DELETE	/posts/{post_id}
책갈피 삭제	DELETE	/bookmark/{post_id}
책갈피 추가	POST	/bookmark/{post_id}
좋아요 추가	POST	/likes/{post_id}/like
좋아요 해제	PATCH	/likes/{post_id}/unlike

Hashtags

표

Aa 이름	Method	URI
해시태그 추가하기	POST	/hashtags?name={hashtag_name}
해시태그 정보 가져오기	GET	/hashtags/{hashtag_id}
해시태그 삭제하기	DELETE	/hashtags/{hashtag_id}

API 명세

Post & Comment Server

Aa 이름	Method	URI	참고	request	response
게시글의 댓글 가져오기	GET	/comments/{post_id}			status
댓글 삭제하기	DELETE	/comments/{post_id}/{comment_id}			status: ok, pending, canceled
댓글 추가하기	POST	/comments/{post_id}	댓글, 대댓글 추가	content: required replied_to_comment_id:	id, from(user), text, created_at
대댓글 보기	GET	/comments/{post_id}/{comment_id}	parent_comment_id에 달린 대댓글 조회		parent_pk, user_pk, child_comments
좋아요 추가	POST	/comments/like/{comment_id}			status, like
좋아요 해제	PATCH	/comments/unlike/{comment_id}			status, like

API 명세

Story Server

Aa 이름	Method	URI	참고	request	response
스토리 가져오기	GET	/stories/{story_id}	스토리 1개를 가져온다.		id, user_pk, image_url, created_date, regrammed_user_pk
스토리 작성하기	POST	/stories/new	스토리를 생성한다.	id, user_pk, image_url, created_date, regrammed_user_pk	200 - OK, 400 - Bad Request
스토리 삭제하기	DELETE	/stories/delete/{story_id}	스토리를 삭제한다.		200 - OK, 400 - Bad Request
좋아요 추가	POST	/stories/likes/like		id, likes_user_pk	200 - OK, 400 - Bad Request
좋아요 해제	POST	/stories/likes/unlike		id, likes_user_pk	200 - OK, 400 - Bad Request
user tag 추가하기	POST	/stories/add-user-tag	user tag를 추가한다.	story_pk, tagged_user_pk, location_x, location_y	200 - OK, 400 - Bad Request

JIKJI

API 명세

Graph Server

Aa 이름	Method	URI	참고	request	response
팔로우 추가하기	POST	/follow/add		follower_id, followed_id	200 - OK, 400 - Bad Request
팔로우 삭제하기	POST	/follow/delete		follower_id, followed_id	200 - OK, 400 - Bad Request
유저 팔로우 내역 조회	GET	/follow/{id}			follower_id, followed_id
팔로우 수 조회	GET	/follow/count/{id}			count

API 명세

Media Server

Media API

Aa 이름	태그	Description	request
api/media	POST	미디어를 등록한다.	file (IMAGE or VIDEO), username

Media API Response Data

Aa 이름	Description
media_name	미디어 이름
url	미디어의 URL
media_type	IMAGE, VIDEO 타입

API 명세

Notification Server

Notifications

표

Aa 이름	Method	URI	참고	request	response
알림 전송	POST	/notifications/push-notice	사용자에게 알림을 전송한다	served_user_pk, service_type, service_pk	200 - OK, 400 - Bad Request

API 명세

Chat Server

채팅방

Aa 이름	Method	URI
채팅방 개설	POST	/chat/{userEmail}
채팅방 나가기	POST	/chat/leaveChattingRoom

채팅

Aa 이름	태그	URI
채팅 보내기	Message	/send/{toUser}

API 명세

Search Server

표

Aa 이름	태그	URI
검색어 조회	GET	/search?keyword={keyword}
+ 새로 만들기		